THE STATISTICAL-MECHANICS OF LEARNING A RULE

WATKIN, TLH; RAU, A; BIEHL, M

Published in:
Reviews of Modern Physics

DOI:
10.1103/RevModPhys.65.499

The statistical mechanics of learning a rule

Timothy L. H. Watkin* and Albrecht Rau†

Department of Physics, University of Oxford, Oxford OX1 3NP, United Kingdom

Michael Biehl

Physikalisches Institut, Julius-Maximilians-Universität, Am Hubland, D-8700 Würzburg, Germany

A summary is presented of the statistical mechanical theory of learning a rule with a neural network, a rapidly advancing area which is closely related to other inverse problems frequently encountered by physicists. By emphasizing the relationship between neural networks and strongly interacting physical systems, such as spin glasses, the authors show how learning theory has provided a workshop in which to develop new, exact analytical techniques.

CONTENTS

*Present address: St. Johns College, Cambridge CB2 ITP, United Kingdom.

†Present address: Patentanwälte Schwabe, Sandmair, Marx, Stuntzstr.16, D-8000 München 80, Germany.

If a thing is worth doing, it is worth doing badly.

Chesterton, 1910

I. INTRODUCTION

A superb example of the power and versatility of statistical mechanics is the field of neural networks, to which the tools of physics have been applied with such spectacular success. Here we describe what is perhaps the most interesting area for nonspecialist physicists, the learning of a rule by a network, a problem which not only has direct analogies to the physics of strongly interacting complex systems, but is also related to the inverse problems of imaging and generalized computation. Statistical mechanics is continuing to solve problems which have been known to engineers for many years, and we expect the new tools and insights developed in this field to make a considerable contribution to conventional physics.

In a sense the theory of neural networks has come full circle. Neural networks were originally proposed as an alternative to conventional computers to perform the task of processing input data into output data (McCulloch and Pitts, 1943). Here we call the input data a *question* and the corresponding desired output data the *answer,* where the answer is derived from the question by a *rule.* Whereas a conventional computer consists of a single processor performing complicated functions of its reservoir of data, a neural network instead has many processors, the *nodes* or *neurons* (we use these words interchangeably), connected together as, for example, in Fig. 1. The output of a neuron is called its *state* and is sent to be an input to the other nodes; thus node 1 in Fig. 1 receives inputs from neurons 2 and 4 and sends its output to be an input to nodes 2 and 3. Each node performs a simple function of its inputs, and the hope is that since all these processors will be working at once, the whole computation will be performed very quickly. This is the foundation of *parallel distributed processing* or *connectionism* (Rumelhart and McClelland, 1986). Engineers were left with the considerable task of deciding which connections between neurons should be made and how neurons should be programmed.

Hopfield (1982) realized that if the state of each node takes just two values, if certain stringent conditions are placed upon the details of the connections and the function of its inputs which each node performs, and if neurons change their states one by one in a random order, then the dynamics of the whole network can be investigated using statistical mechanics, because it is possible to define an *energy* for the network which obeys detailed balance. The functions which the neurons perform may be adjusted (within the constraints of the Hopfield model) so that prechosen configurations of the network are fixed points of the network dynamics; it is speculated that these fixed points are related to *memories* in real, biological networks—hence our use of the word "neuron."

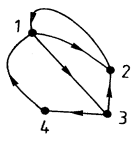

FIG. 1. A network of four neurons.

Although storage of memories continues to be an area of active interest (for reviews see Amit, 1989; Geszti, 1990; Ritter *et al.*, 1990; Domany *et al.*, 1991; Müller and Reinhardt, 1991; Peretto, 1991; Weisbuch, 1991), it is only loosely connected to computation, the subject of this paper, since Hopfield's network does not take input data (except as the initial states of all the nodes), and since the networks we shall study obey neither Hopfield's constraints nor any of the subsequently devised alternative sets of constraints which make an analysis of the fixed-point structure possible (Peretto, 1984; Derrida *et al.*, 1987; Anlauf and Kuhlmann, 1992; Watkin and Sherrington, 1992). Instead our subject will be networks in which some nodes, the *input nodes,* are fixed in a configuration corresponding to a question, and other nodes, the *output nodes,* are naturally set by the dynamics of the network to a configuration of states corresponding to the question's answer. This is the architecture used in most engineering applications.

The natural method of programming a conventional sequential computer is a list of instructions, which is a severe limitation in problems such as speech recognition, for which the rule is not exactly known. Schemes exist, however, to teach a neural network from *examples,* that is, pairs of questions and correct answers. In a sense this is an imaging problem, because an unknown rule is reconstructed from the data it generates, but our intention is not specifically to infer the rule but to answer new questions correctly. Learning in this fashion is called *supervised learning,* because it requires a *teacher network* providing the correct answers to the questions from which our *student network* must learn.

Neural networks have achieved many successes, but their underlying theory remains notoriously difficult to formalize. Even the most advanced statistics which has been applied, VC theory (Sec. II.E), has been unable to do much more than place weak bounds on their success, and then only for the simplest networks. As a result, most practical network research is carried on in a trial-and-error fashion. Frequently "tricks" are tried to improve learning efficiency, but these tricks are usually motivated either by an appeal to intuition or by using only the simplest principles of information theory.

During the past few years, however, a theory of learning has been proposed which uses *statistical mechanics.* Instead of weak bounds, the new theory is able to make

exact predictions in idealized problems and has thus solved problems which have been outstanding in the machine-learning community for many years. More importantly, statistical mechanics gives real insight into practical applications. New techniques for learning may be motivated by exploiting advanced statistical physics, particularly that of spin glasses.

This article summarizes the statistical mechanics of learning.

Section II defines the concepts of a network and a rule more precisely and thus sets up the problem to which statistical mechanics is applied in subsequent sections. As an illustration, we develop, in parallel, an example of a simple network, the *perceptron* (Rosenblatt, 1962), on which the majority of analysis has centered, and fully analyze learning from examples by a simple algorithm. In this case geometrical insight alone may avoid a complex statistical-mechanics formulation. We then explain the recent information-theoretic *Bayesian* approach to learning. Although we keep Bayesian sections separate from the rest of Sec. II, we believe that Bayesian insights are particularly important. They have an intimate connection to the most recent engineering advances. Bayesian theory allows us to define in principle the optimal way in which generalization can be performed (Sec. II.F.2) and the optimal way to train a neural network (Sec. II.G.2).

Section III shows that statistical mechanics is a useful tool for analyzing learning for two different reasons. One is that many algorithms are stochastic and correspond exactly to Langevin or Glauber dynamics on a noisy energy landscape (Sec. III.A.2). The behavior of these algorithms may be analyzed using statistical mechanics, and this allows us to infer how other algorithms would behave. Another reason for using statistical mechanics is that learning is essentially a problem of statistical interference and fits naturally, as we shall see in Sec. III.A.3, into the same mathematical framework.

The examples we have to learn from will inevitably be chosen from some distribution and, once they are, will constitute a static "quenched" disorder. The effects of quenched disorder have been studied in many areas of condensed-matter physics, and a mean-field theory has been developed to analyze them. Section III.A develops a simple, but exact, neural network example, which gives a flavor of the techniques involved.

Section III.B shows how insights into the problem let us design more sophisticated learning algorithms for a perceptron. In Sec. III.C, we summarize a number of exactly solved models, whose features have turned out to be generic. In some, a rich structure of metastable states may exist, just as in other disordered systems, and this is of vital interest for the convergence times of algorithms.

As Sec. IV shows, it is sometimes possible to analyze completely the approach to the minimum of the energy landscape: exact dynamical solutions in noisy, nonequilibrium situations.

Further richness appears in practical applications, as emerges in Sec. V. If constraints exist on the complexity of the network with which we are allowed to learn and if a rule is too complicated to be learned exactly within these constraints, then it will generally be impossible to answer correctly all the examples we are given. Incompatible examples lead to frustration, in direct analogy to spin glasses, but insight may help us to avoid its unfortunate consequences (Sec. V.A). We study the case of noisy examples in Sec. V.B. It is possible also to conceive of an *intelligent student* who asks questions depending upon what has already been learned; this interrogative approach to information extraction has clear implications for other inverse problems and remains an area of active study (Sec. V.C). Conversely, we describe a situation in which examples of the rule are unlimited and learning must be performed with as little computational effort as possible (Sec. V.D). We consider, more generally, how prior information about the type of rule being learned lets us intelligently design neurons to perform a more complicated function (Sec. V.E). Section V.F shows that in some realistic problems, it may be worthwhile to choose a definition of energy so as to *sculpt* the energy surface on which the network evolves.

Section VI describes how much of this theory applies to multilayer networks (MLN), which are the type already in common engineering use. It is a measure of how quickly this subject is developing that most of the statistical mechanics described in this section appeared in preprint form while our article was being written. Many of the open problems we listed in the first edition had been solved in time for the second edition.

Section VII discusses the usefulness of the statistical-mechanics approach to learning and points out the directions in which new statistical mechanics is expected to lead.

In summary, the structure of the physical ideas presented in this paper is as follows: Sec. II, formulation of the problem; Sec. III, relation of the problem to that of spin glasses, introduction to the *replica method* and its physical significance; Sec. IV, analysis of disordered dynamics using the eigenvalue spectrum of a random matrix and ergodicity breaking without *replica symmetry breaking;* Sec. V, generalization of the basic model, including information extraction by interrogation and design of energy landscapes; Sec. VI, analysis of learning in more realistic multilayer networks; Sec. VII, the areas in which new physical insight is required.

Although this article is written for general physicists, its secondary purpose is to stimulate new research. There still exists, for example, great scope for new work in more complicated networks where, as shown in Sec. VI, although certain simple results have been derived, there is so far no general insight. In these problems it is to be expected that systematic procedures will be developed to avoid the difficulties of the empirical approaches adopted by most previous researchers.

II. THE FORMALISM OF LEARNING

In this section the concepts of a neural network and a rule are carefully defined. We show how naturally the

two may be combined and develop, in parallel, a straightforward example, *perceptron* (Rosenblatt, 1962; Minsky and Papert, 1969).

A. The definition of a network

1. Formal definition

A neural network, in its most general form, is a system of *neurons* or *nodes,* each of which is associated with a real numerical value, the *state* of the neuron. Frequently the values are restricted to ± 1, so that each node is like an Ising spin (Ising, 1925).

Connections exist between the neurons so that the state of one neuron may influence the state of the others. Consider, for example, the network of six nodes in Fig. 2(a), where neurons are labeled by the numbers 1–6. Neuron 4 receives *inputs* from nodes 1 and 2 (that is, the information about which state nodes 1 and 2 are in) and sends information about its own state, its *output* as an input to nodes 1 and 5. Notice that loops exist in this network: 1 sends its output to 4, which sends its output back to 1; such a network is called *recurrent.* Figure 2(b), however, shows a *nonrecurrent* network: no loops of information exist, although information may still pass from one point to another by different routes, e.g., from 3 directly to 6 or via node 4.

The network in Fig. 2(b) is also *layered* (Domany *et al.*, 1986); neurons 1, 2, and 3 form the first layer; neurons 4 and 5 are layer 2; and neuron 6 is in layer 3. Each layer can only influence higher layers or the same layer. Notice that a layered network may still be recurrent; in the net in Fig. 2(c), neuron 4 sends signals to 5, which signals back. The first layer of a layered network is often called the *input layer,* and the last is the *output layer.* Layers in between are frequently called *hidden layers.*

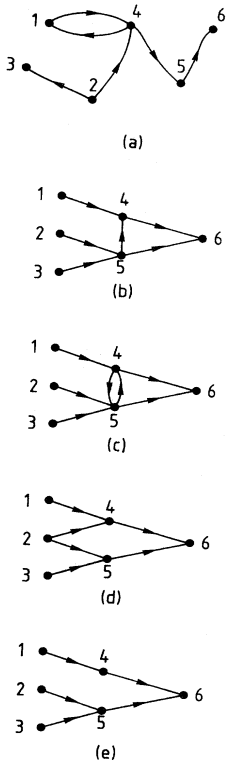

FIG. 2. Networks of six neurons: (a) recurrent; (b) layered; (c) layered and recurrent; (d) feed-forward; (e) treelike.

A *feed-forward* network [for example, that in Fig. 2(d)] does not even allow signals between neurons in the same layer. It does, however, allow information to pass from the first layer to the last by different routes. A network is called *treelike* if its structure is like that in Fig. 2(e): information about the state of neurons in layer 1 may reach node 6 in layer 3 in exactly one way.

How does a neuron react to its inputs? Consider a neuron which we shall call A, whose state we call S_A, and which receives inputs from the set of k neurons we shall call $\{A_j\}$, $j=1,\ldots,k$. The states of these neurons we label $\{S_{A_j}\}$. Generally we might allow the state of A to be set to any function F_a of the inputs it receives, that is,

$$S_A \mapsto F_A(\{S_{A_j}\}) , \tag{2.1}$$

where we have pointed out that the function may depend upon A. Once the initial state of every neuron has been set and the function it performs is known, we may move around the network updating neurons according to their function. The dynamics of such networks has been studied intensively (for a review see Amit, 1989). For a feed-forward network, however, we may consider the states of the neurons in the input layer to be fixed, use the functions (2.1) to set the states of neurons in layer 2, and so on until the state of every neuron in the network is determined. It is this second case which we shall be considering in this paper.

Usually the functions a neuron may perform are restricted to the form

$$S_A \mapsto F_A\left[\sum_{j=1}^{k} J_{A,A_j} S_{A_j}\right] . \tag{2.2}$$

That is, the state is set to a function (which may depend upon A) of the weighted sum of the inputs to A, where the set $\{J_{A,A_j}\}$ are the weights. Setting these weights is the principal task of learning theory. This form of updating function was originally motivated by biological models in which the weight vectors represented the strengths of synoptic connections between neurons (Hebb, 1949); so functions of the form Eq. (2.2) are called "synaptic emulation."

Usually we shall assume that there is only one neuron in the output layer, node o, whose state is S_o. Similarly, we shall assume that there are N nodes in the first layer, labeled by i, where $i=1,\ldots,N$, so that the states of the input nodes are the components of an N-vector $\mathbf{S}$. The function which the network performs on its input to produce its output is called $\mathcal{N}$. Thus $S_o=\mathcal{N}(\mathbf{S})$.

We shall call the space of all networks we are allowed to build—including the number of nodes, their connections, the functions each node performs, and the order in which they are updated—the *student space.*

2. Example: the perceptron

Consider the network in Fig. 3(a), the *perceptron* (Rosenblatt, 1962). Since there are no hidden layers, we can simplify our notation and write the function performed by the network as

$$S_o = f\left[\sum_i J_i S_i\right] , \tag{2.3}$$

where the $\{J_i\}$ are the weights between the input nodes and the output node. Here the sum is over all N values of i, and this notation is used for the perceptron throughout this paper. If the states of neuron o are restricted to ± 1, we might choose the function f so that

$$S_o = \mathrm{sgn}(\sqrt{N}\,\mathbf{J}\cdot\mathbf{S} - \psi) . \tag{2.4}$$

Here ψ is a constant *threshold* and we have defined the scalar product of two N-vectors $\mathbf{a}$ and $\mathbf{b}$ as $\mathbf{a}\cdot\mathbf{b} \equiv \sum_i a_i b_i / N$. Equation (2.4) contains a factor of $\sqrt{N}$ for a convenience which will appear below. The variables ψ and $\{J_i\}$ completely define a perceptron's action; so $\mathbf{J}$ is called the *perceptron vector.*

This system, in which S_o is restricted to ± 1, is called a *binary perceptron.* Choosing $f(x)=x$, so S_o may take any real value, makes the system a *linear perceptron* (Hertz *et al.*, 1991). If each component of the vector $\mathbf{J}$ can take any real value (such that the whole $\mathbf{J}$ vector has a constant normalization, as explained below), we call the perceptron *spherical;* if it is restricted to ± 1, the resulting perceptron is called *Ising.* Thus, for example, an *Ising binary perceptron* has a ± 1 output, and the components of the $\mathbf{J}$ are also ± 1. Our article will use this nomenclature consistently, although within the literature there exists some confusion of terminology. The spherical and Ising constraints are the upper and lower extremes of $\mathbf{J}$-space freedom; of the two the Ising case is perhaps the most instructive, because, in a real network, engineering restrictions are likely to lead to quantized couplings. Kurchan and Domany (1990) have also analyzed more general constraints on the $\mathbf{J}$-space.

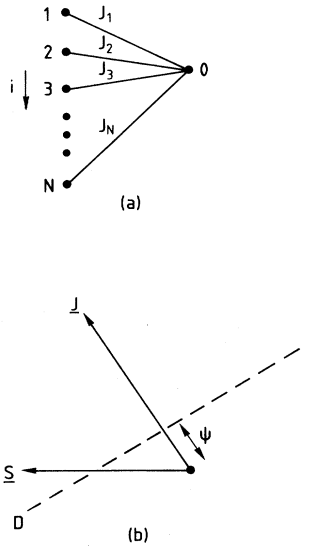

FIG. 3. The perceptron: (a) architecture of N input nodes labeled by i, sending signals weighted by J_i to output o; (b) section of the N-dimensional input space, showing the perceptron vector $\mathbf{J}$ perpendicular to hyperplane $\mathcal{D}$. $\mathcal{D}$ is displaced by ψ from the origin $\mathcal{O}$, and its projection into the plane of the diagram is shown as the dashed line. Input $\mathbf{S}$ gives output $+1$.

Notice that the output S_o of the binary perceptron is invariant under the transformations

$$J_i \to \lambda J_i \quad \text{for all } i , \tag{2.5}$$

$$S_i \to \nu S_i \quad \text{for all } i , \tag{2.6}$$

$$\psi \to \nu\lambda\psi , \tag{2.7}$$

for any positive values of λ and ν. These are the *gauge freedoms* of the perceptron. We normally fix the λ gauge so that

$$\mathbf{J}\cdot\mathbf{J} = 1 , \tag{2.8}$$

and take ψ to be positive or zero. The space of possible spherical $\mathbf{J}$'s is therefore the unit N-sphere, so that in this case the student space is the product of the unit N-sphere and the set of positive real numbers ψ. If the perceptron is Ising, then $\mathbf{J}$ lies on one of the corners of a unit N-cube. Similarly we shall fix the ν gauge by taking every component of every question to be drawn from a distribution with $\langle S_i^2 \rangle = 1$. $\mathbf{S}$ vectors, in which every component is drawn randomly and independently from a distribution which is independent of $\mathbf{J}$, will have an overlap with $\mathbf{J}$ which is the result of a random walk. Thus $\sqrt{N}\,\mathbf{J}\cdot\mathbf{S}$ is of order 1, which explains the factor of $\sqrt{N}$ in Eq. (2.4).

An easy way to visualize the action of a perceptron is from a schematic diagram, Fig. 3(b), of the N-dimensional *input space,* which is all the possible configurations of the nodes in the input layer. The vector $\mathbf{J}$ is shown in this space and is perpendicular to the $(N-1)$-dimensional hyperplane $\mathcal{D}$, which is displaced from the origin $\mathcal{O}$ by a distance ψ. Only input configurations, such as $\mathbf{S}$, which fall on the same side of $\mathcal{D}$ as the direction of $\mathbf{J}$, have a positive $\sqrt{N}\,\mathbf{J}\cdot\mathbf{S} - \psi$ and will cause S_o to be set to $+1$. Usually (and throughout the rest of this paper) the perceptron is further restricted to $\psi = 0$, so that $\mathcal{D}$ passes through the origin and the student space is just the unit N-sphere.

Neural networks have frequently been studied in the context of storing *memories—that is, random, prechosen configurations of the states of all* neurons in the network—by designing the weight strengths so that the memories are fixed points of the network's dynamics (Amit, 1989). In a highly recurrent network this is a considerable task; but in a feed-forward network, such as a perceptron, it simply means that if the states of the input

neurons are set to the corresponding digits of the memory configuration, then the dynamics of the network must set the states of the output neurons equal to the rest of the memory. As explained in Sec. II.C, storage of memories is very different from learning a rule.

B. Definition of a rule

1. Formal definition

A rule associates a question and an answer. Here the question is a vector of (usually) high dimension; the answer is a vector (usually of much smaller dimension) which is determined by the question. In most of this paper we shall assume that answers are one dimensional, although multidimensional answers are a straightforward extension. One example of this formalism would be the recognition of speech: the high-dimensional frequency spectrum of a slice of time must be transformed into one of a list of words. Writing a question as $\mathbf{r}$ and the answer as $\mathcal{R}$, we say that they are associated by rule V so that

$$\mathcal{R}=V(\mathbf{r}) . \tag{2.9}$$

Thus $\mathbf{r}$ is an element of the *question space,* $\mathcal{R}$ of the *answer space,* and V of the *rule space.*

For the purpose of framing statistics, we shall suppose that the rule we have to learn has been drawn randomly from some statistical distribution $P(V)$. Thus $P(V)$ is the *measure* of the space of rules. Of course, in practice, $P(V)$ is unknown, even hypothetical; but this usually turns out not to be a problem. Some results can be found which apply to *any* underlying rule (Sec. VI.D); many others are true for all rules of given "complexity." There is an alternative Bayesian interpretation of $P(V)$, which is discussed in Sec. II.F.1.

Unfortunately, many neural network papers use the word "rule" with a different meaning: as an algorithm for building a network. We shall use our meaning of "rule" throughout this article.

Suppose that we have p example questions $\{\xi^\mu\}$, $\mu=1,\ldots,p$, for each of which we know the answer $\{\xi_o^\mu\}$. The known question-and-answer examples of the rule form the *training set.* Each example places a constraint upon the places in rule space where the true V must lie, so that after seeing p examples the region in which V must lie is reduced to $\mathcal{V}^{(p)}$. In artificial intelligence literature, $\mathcal{V}^{(p)}$ is called the *version space* (Mitchell, 1982).

2. Example: a linearly separable rule

We define a *linearly separable* rule as one of the form

$$\mathcal{R}=V(r)\equiv\mathrm{sgn}(\sqrt{N}\,\mathbf{B}\cdot\mathbf{r}-\phi) , \tag{2.10}$$

where $\mathbf{B}$ is any vector. Here $\mathcal{R}=\pm1$. By analogy with Fig. 3(b), we can draw Fig. 4(a), showing a schematic two-dimensional section of the N-dimensional space containing vector $\mathbf{B}$, which is normal to plane $\mathcal{C}$ displaced by ϕ from the origin $\mathcal{O}$. Questions falling onto the same side of $\mathcal{C}$ as the positive $\mathbf{B}$ direction, such as question $\mathbf{r}_1$, will be answered by $+1$, and others, such as $\mathbf{r}_2$, will be answered -1, so that question space is divided by a plane (hence the name linearly separable).

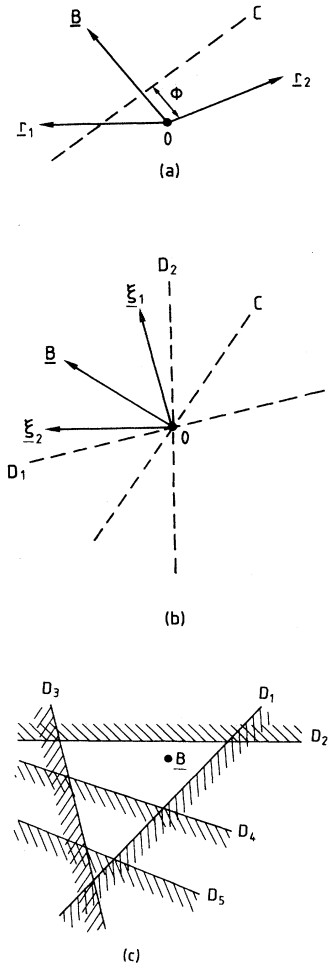

FIG. 4. Learning a linearly separable rule: (a) A section of the N-dimensional question space, containing $\mathbf{B}$, which is perpendicular to hyperplane $\mathcal{C}$. The projection of $\mathcal{C}$ into the plane is marked dashed and it is displaced from the origin by distance ϕ. (b) Both examples ξ_1 and ξ_2 are answered $+1$, so $\mathbf{B}$ must lie between planes $\mathcal{D}_1$ and $\mathcal{D}_2$. (c) Schematic "top view" of the situation, looking along the negative $\mathbf{B}$ direction. Planes $\mathcal{D}_1$–$\mathcal{D}_4$ constrain $\mathbf{B}$ to the version space. Plane $\mathcal{D}_5$ adds no information.

Except in Sec. V.A.1, we shall take $\phi=0$, so that only the direction, and not the magnitude, of $\mathbf{B}$ is relevant; we conventionally assume that the magnitude of $\mathbf{B}$ is normalized to 1, ($\mathbf{B}\cdot\mathbf{B}=1$), so that the rule space is the unit N-sphere. In the rule space of linearly separable rules, $\mathbf{B}$ defines the rule; so $P(V)$ is equivalent to $P(\mathbf{B})$, the statistical distribution of $\mathbf{B}$ over the surface of the sphere.

The linearly separable rule, Eq. (2.10), without a threshold ($\phi=0$) is defined by $\mathbf{B}$, an element of the N-dimensional rule space. Since there are only two possible

output results, $\mathcal{R}=\pm 1$, the answers to examples in the training set are $\xi_o^\mu=\pm 1$. Referring to Fig. 4(b), questions ξ^1 and ξ^2, perpendicular to the hyperplanes $\mathcal{D}_1$ and $\mathcal{D}_2$, are also marked; both answers, ξ_o^1 and ξ_o^2, are $+1$. Thus from the answer to these questions we can infer that **B** lies in the region between $\mathcal{D}_1$ and $\mathcal{D}_2$. This region is the version space.

Figure 4(c) is a schematic diagram of the situation looking towards the origin along the negative **B** direction. The four planes $\mathcal{D}_1$, $\mathcal{D}_2$, $\mathcal{D}_3$, and $\mathcal{D}_4$ are constraints on **B** from examples 1–4, and each is shown forcing **B** to lie on the unshaded side of the plane. Notice that plane $\mathcal{D}_5$, from some example ξ^5, adds no new information, since planes $\mathcal{D}_1$–$\mathcal{D}_4$ already constrain **B** to an area agreeing with example 5: example 5 has not reduced the version space.

C. Learning a rule with a network

1. The problem

The purpose of learning is to design a network such that if the states of nodes in the input layer are set equal to a question, the states of the nodes of the output layer will become equal to the correct answer. For this problem to be well posed, the possible states of input and output neurons must be the same as those of the components of the questions and answers, respectively.

This problem is quite different from that of storing memories (Sec. II.A) because questions are related to answers by a rule (whereas memories are usually uncorrelated), and because we are trying to deduce the answers to *new* questions, rather than just recall the answers to old ones. The learning of rules has many more engineering applications.

There are two distinct tasks of learning: One is, knowing rule V, to construct a network which will reproduce it. The other is to design a network just from question-and-answer pairs. So far it is this second case which has received the attention of statistical mechanics. This is called *supervised learning,* because it requires a *teacher,* knowing the rule, which gives the correct answer to the example questions.

Why does this work? Why, indeed, should teaching a network some examples help it predict the answers to others? The answer is that it will only if the rule space and the student space perform similar functions of their inputs, so that the constraint of correctly answering the training set forces the student network to resemble increasingly the teacher. Otherwise, learning from examples will be completely unsuccessful, as we shall see in Sec. V.A.3, when a perceptron tries to learn the parity problem. Notice that it is the restrictions on the student space which make learning possible. We shall see a fascinating implication of this observation in Sec. VI.C.1.

A rule for which a network in student space exists which gives the correct answer to all possible questions is called *learnable.* Conversely, a rule which no network in the student space can learn exactly is called *unlearnable.*

2. Formalism

As above, the function performed by our network is characterized by the symbol $\mathcal{N}$. In order to measure the deviation of the network output $\mathcal{N}(\mathbf{S})$ from the teacher rule $V(\mathbf{S})$, we introduce an arbitrary error measure $e[\mathcal{N}(\mathbf{S}),V(\mathbf{S}),\mathbf{S}]$, which is zero if teacher and student agree on the answer to **S** and larger than zero when they disagree. The most obvious choice for e is the binary measure ($e\in\{0,1\}$). The error measure $e[\mathcal{N}(\xi^\mu),V(\xi^\mu),\xi^\mu]$ may be considered a function of the discrepancy of $\mathcal{N}$ on the μth question with respect to the correct answer ξ_o^μ, in which case, knowing the question and answer and whether $\mathcal{N}$ agrees, e is *observable*; alternatively, e can be considered the disagreement on the μth example between $\mathcal{N}$ and the unknown rule V.

Similarly we may define *extensive energies,* where extensive means scaling with the number of examples, which we shall denote using the letter E. If energies are *observables*—that is, not functions of the unknown rule—they may be used in algorithms. One such energy we shall use frequently is the training energy

$$E_t(\mathcal{N},\{\xi^\mu\},\{\xi_o^\mu\})=\sum_\mu e(\mathcal{N},\xi_o^\mu,\xi^\mu)\ , \tag{2.11}$$

which is a measure of the number of examples in the training set which $\mathcal{N}$ answers wrongly.

However, the quantities we shall mainly want are *expectation values* of the performance, which will be calculated naturally in terms of variables which are functions of V, and therefore not observable. These we shall denote by ϵ. For example, the average disagreement between $\mathcal{N}$ and V is the *generalization function*

$$\epsilon_f(\mathcal{N},V)\equiv\int d\mu(\mathbf{S})e(\mathcal{N},V,\mathbf{S})\ , \tag{2.12}$$

where $d\mu(\mathbf{S})$ denotes the normalized distribution from which the questions **S** are chosen, so that integrating over it simply means performing the average over the distribution of questions. $\epsilon_f(\mathcal{N},V)$ is the measure of the quality of network $\mathcal{N}$ in reproducing rule V.

We shall shortly consider algorithms to construct networks. Some algorithms give a particular $\mathcal{N}$, others give an $\mathcal{N}$ drawn from some distribution. In either case we shall denote the average over the networks produced by the algorithm by $\langle\cdots\rangle_{\mathcal{N}}$. If the underlying rule is V, the network we build using the algorithm has "badness"

$$\epsilon_g(V,\{\xi^\mu,\xi_o^\mu\})=\langle\epsilon_f(\mathcal{N},V)\rangle_{\mathcal{N}}\ . \tag{2.13}$$

Notice that this error measure is still a function of (i) the unknown underlying rule and (ii) the training set. Since the answers in the training set are defined by the questions, this is equivalent to saying that Eq. (2.13) is a function of the underlying rule and the *questions* of the training set: two independent forms of disorder.

It turns out, however, that if the questions and rules are drawn from any of a large class of distributions, then the value $\epsilon_g(V,\{\xi^\mu\})$ is *self-averaging,* which means that almost any realization of the underlying rule and any set of questions give the same result. Thus for a given distribution of training set questions, the quantities of interest, such as ϵ_g, are functions only of the algorithm generating $\mathcal{N}$, the *number* of examples from which we have to learn, and the *distribution* from which the questions are taken.

The value can be found by calculating

$$\epsilon_g = \langle \langle \epsilon_g(V,\{\xi^\mu\}) \rangle_V \rangle_\xi = \langle \langle \langle \epsilon_f(\mathcal{N},V) \rangle_{\mathcal{N}} \rangle_V \rangle_\xi , \tag{2.14}$$

where we are using the exterior brackets $\langle \cdots \rangle_V$ to indicate the average over the space of all rules, and the brackets $\langle \cdots \rangle_\xi$ to indicate the average over sets of *questions.* That is, these brackets together denote the average over realizations of the *quenched disorder.*

Similarly we may define the average training error as

$$\epsilon_t = \left\langle \left\langle \frac{1}{p} \langle E_t(\mathcal{N},V,\xi^\mu) \rangle_{\mathcal{N}} \right\rangle_V \right\rangle_\xi . \tag{2.15}$$

In Sec. II.D we give a simple example of the use of this formalism. In Sec. III.A.2 we shall show that statistical mechanics is a natural way to calculate quantities such as ϵ_g for more advanced algorithms.

D. Hebbian learning with a perceptron—an example

It is clear that a perceptron, Eq. (2.4), can learn a linearly separable rule, Eq. (2.10), without a threshold, since if we set $\psi=0$ and $\mathbf{J}=\mathbf{B}$, then $S_o=V(\mathbf{S})$. Learning can be illustrated from Fig. 5(a), a sketch showing the two-dimensional subspace of the input space containing vectors $\mathbf{B}$ and $\mathbf{J}$ [i.e., the $(\mathbf{B}-\mathbf{J})$ plane]. The vectors $\mathbf{B}$ and $\mathbf{J}$ both lie on the surface of the unit N-sphere and are normal to the $(N-1)$-dimensional hyperplanes $\mathcal{C}$ and $\mathcal{D}$, respectively; they are at an angle θ (i.e., $\mathbf{B}\cdot\mathbf{J}=\cos\theta$). The projectrons of $\mathcal{C}$ and $\mathcal{D}$ into the $(\mathbf{B}-\mathbf{J})$ plane are labeled lines. From Eq. (2.10), the correct answer to a question is $+1$ if its projection into the space of $\mathbf{B}$ and $\mathbf{J}$ lies on the same side of plane $\mathcal{C}$ as $\mathbf{B}$, and -1 otherwise. The perceptron will wrongly answer questions whose projection into the plane falls into areas E or F.

Consider questions whose components are taken independently from a distribution whose first moment is zero and second moment is $\mathcal{O}(1)$; for example, each component might be randomly ±1. If, similarly, each component of $\mathbf{B}$ is also chosen from such a distribution and $\mathbf{J}$ has similar properties, then we may treat examples as falling randomly onto the N-sphere, and their projections into the $(\mathbf{B}-\mathbf{J})$ plane have a random direction away from origin. The generalization function, the chance that such questions are answered wrongly, is thus the proportion of the $(\mathbf{B}-\mathbf{J})$ plane in areas E and F,

$$\epsilon_f(\mathbf{J},\mathbf{B}) = \frac{2\theta}{2\pi} = \frac{1}{\pi}\cos^{-1}(\mathbf{B}\cdot\mathbf{J}) , \tag{2.16}$$

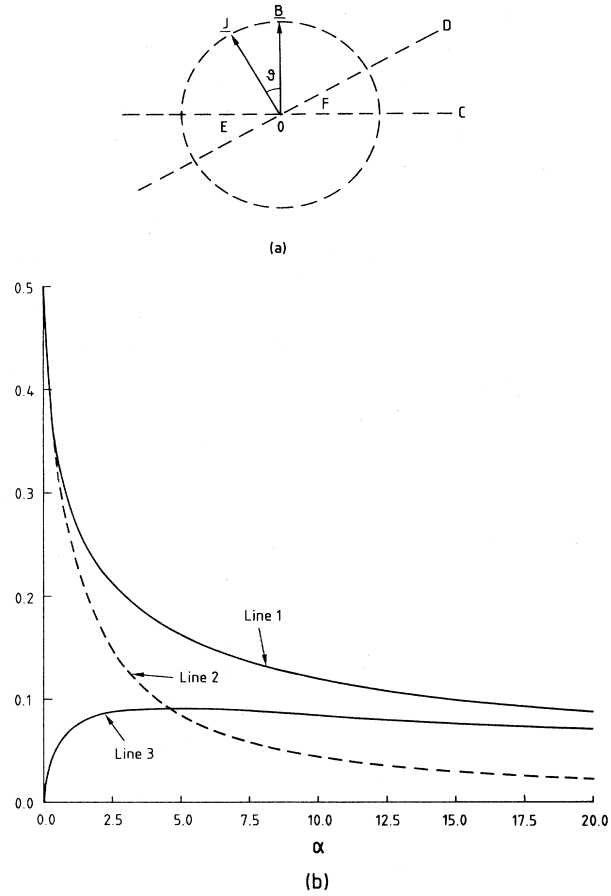

FIG. 5. Learning a linearly separable rule with a perceptron: (a) The $\mathbf{B}$-$\mathbf{J}$ plane in question space. $\mathbf{J}$ and $\mathbf{B}$ are perpendicular to hyperplanes planes $\mathcal{D}$ and $\mathcal{C}$, respectively. $\mathcal{D}$ and $\mathcal{C}$ intersect at the origin $\mathcal{O}$, and two of the volumes enclosed by them are labeled E and F. (b) Line 1 shows the generalization error of the Hebb algorithm against α, and line 2 shows the optimal result which can be achieved (Sec. III.C.1). Line 3 shows the Hebb algorithm's training error.

where we have used the fact that the perceptron is specified by $\mathbf{J}$ to write $\mathbf{J}$ for $\mathcal{N}$, and the fact that V is specified by $\mathbf{B}$ to write $\mathbf{B}$ for V.

We shall now illustrate a simple way in which a spherical perceptron may be trained to learn a linearly separable rule. We do this using a training set $\{\xi^\mu,\xi_o^\mu\}$, consisting of p examples of the rule [that is, $\xi_o^\mu=\mathrm{sgn}(\sqrt{N}\,\mathbf{B}\cdot\xi^\mu)$ for $\mu=1,\ldots,p$]. As in other algorithms, the number of questions required typically scales with N, because $\mathcal{O}(N)$ bits of information are required approximately to constrain the $(N-1)$ degrees of freedom of $\mathbf{B}$, so that defining α from $p\equiv\alpha N$, the interesting limit is α remaining constant as $N\to\infty$.

Vallet (1989) set $\mathbf{J}$ by the Hebb algorithm,

$$\mathbf{J} \equiv \frac{1}{\gamma\sqrt{N}} \sum_\mu \xi_o^\mu \xi^\mu , \tag{2.17}$$

where γ is a constant chosen to normalize $\mathbf{J}$. From Figs.

4(a) and 5(a) it is clear that the components of examples in the direction **B** will add constructively to **J**, since Eq. (2.17) means that questions are added to **J** with a sign equal to that of their component in the **B** direction. If we label the distribution of the projection of examples in any random direction **Z** as $x_{\mathbf{Z}} \equiv \sqrt{N}\,\boldsymbol{\xi}\cdot\mathbf{Z}$, then we observe it is a random walk of N steps, so that $x_{\mathbf{Z}}$ is Gaussian distributed,

$$\mathrm{Prob}(x_{\mathbf{Z}}=y)=\frac{1}{\sqrt{2\pi}}\exp(-y^2/2)\ . \tag{2.18}$$

Thus the component of **J** in the **B** direction is $p\langle|x_{\mathbf{B}}|\rangle/(\gamma\sqrt{N}\sqrt{N})=\alpha\sqrt{2}/(\gamma\sqrt{\pi})$. Components of the p questions in the other $N-1$ input space directions add to **J** randomly, as a random walk of p steps of average length $(\langle x^2\rangle)^{1/2}$ in an $(N-1)$-dimensional space, so that the component of **J** perpendicular to **B** is $(p\langle x^2\rangle)^{1/2}/(\gamma\sqrt{N-1})=\sqrt{\alpha}/\gamma$ as $N\to\infty$. Thus the **J** given by Eq. (2.17) is such that the distribution of its overlap with the true **B** is sharply peaked, so that

$$\cos^{-1}(\mathbf{J}\cdot\mathbf{B})=\tan^{-1}(\sqrt{\alpha}/\alpha\sqrt{2/\pi})\ . \tag{2.19}$$

Therefore, from Eqs. (2.14) and (2.16), we can write

$$\epsilon_g=\frac{1}{\pi}\tan^{-1}(\sqrt{\pi/2\alpha})\ . \tag{2.20}$$

This easy derivation was pointed out by Watkin, Rau, Bollé, and van Mourik (1992), and the curve is plotted as line 1 in Fig. 5(b).

Before we have any examples, $\alpha=0$ and the generalization error must be $\frac{1}{2}$, because **B** could be any vector; so **r** might equally well lie on either side of it and the chance that we guess wrongly is $\frac{1}{2}$. As α rises the generalization error for the Hebb algorithm falls to zero. As a measure of the quality of the algorithm, Fig. 5(b) also shows, as dashed line 2, the generalization error which can be achieved by the best possible learning algorithm, as demonstrated in Sec. III.C.1.

The Hebb algorithm is simple, but the **J** it generates has a finite training error, because the noise on the random questions is allowed to interfere: from Eq. (2.17) we can write

$$\boldsymbol{\xi}^{\mu}\cdot J=\frac{1}{\gamma\sqrt{N}}\left[\xi_o^{\mu}+\boldsymbol{\xi}^{\mu}\cdot\sum_{\nu\neq\mu}\xi_o^{\nu}\boldsymbol{\xi}^{\nu}\right]\ , \tag{2.21}$$

where the first term in the parentheses is the signal and the second term is a noise. For those questions for which the noise is greater than 1 (and of the opposite sign to ξ_o^{μ}), **J** will, from Eq. (2.4), give the wrong result. An argument similar to that given above shows that the training error, first derived by Vallet (1989), has a self-averaging value of

$$\epsilon_t=\tfrac{1}{2}-\int_0^{\infty}Du\ \mathrm{erf}\left[u\sqrt{\alpha/\pi}+\frac{1}{\sqrt{2\alpha}}\right]\ , \tag{2.22}$$

where $Du\equiv du\,e^{-u^2/2}/\sqrt{2\pi}$ and $\mathrm{erf}(x)\equiv 2/\sqrt{\pi}\int_0^x dy\,e^{-y^2}=2\int_0^{x\sqrt{2}}Du$; ϵ_t is plotted as line 3 in Fig. 5(b). It rises from zero to a peak at $\alpha\approx 4$, but then tends to ϵ_g, and together they tend to zero as $\alpha\to\infty$ (when $\mathbf{J}\to\mathbf{B}$).

Note that taking the large N limit in the argument above has shown that the distribution of the overlap between **J** and **B** is very sharply peaked. This shows that almost all rules which could have produced the examples—i.e., all the rules in version space—have the same overlap with the Hebb **J**. Thus, as discussed in Sec. II.C, the result is self-averaging. The generalization ability of the Hebb algorithm is a function only of the number of examples asked and their distribution.

From Eq. (2.20), one finds that ϵ_g scales as $1/\sqrt{\alpha}$ as $\alpha\to\infty$. For example, in a network of $N=10^3$ neurons, it would require of the order of $p=10^9$ examples to make the generalization error from the Hebb algorithm $\approx 10^{-3}$; so, clearly, it is worth finding more efficient algorithms. We shall see also, Sec. V.A.3, that the Hebb algorithm may fail to learn much harder rules altogether. Nor is it possible to implement the Hebb algorithm on a perceptron with Ising couplings (which should be used if we knew that the components of **B** were ± 1), except by an unnatural *clipping* operation (van Hemmen, Grensing, Huber, and Kühn, 1988) in which a spherical **J** is forced to obey the Ising gauge constraint by truncating each component of **J** to be equal to its sign. Nor is it obvious what the analog of the Hebb algorithm is for more complicated networks. It is therefore well worth investigating more advanced constructions. On the other hand, Hebb learning is very simple (hence computationally cheap); so we shall investigate some variants of it in Sec. V.D.

How generic are these results? In fact, all known learning schemes have $\epsilon_t<\epsilon_g$, as observed here for Hebb learning; networks do better on the training set than on new examples. In addition, as $\alpha\to\infty$, $\epsilon_t\to\epsilon_g$ in general. Like the algorithms we shall meet in Sec. III, it does not matter whether the components of questions in the training set are ± 1 or are drawn from a Gaussian distribution, provided that the central limit theorem can be applied to the overlap between questions and **B**. However, the rise and fall of ϵ_t is not typical of other algorithms; typically E_t either remains zero (for learnable rules) or rises monotonously.

E. A brief introduction to VC theory

We have now introduced enough tools to make possible a comparison between the statistical-mechanics approach to learning, the topic of this paper, and an alternative approach which is well known in computer science: probably almost correct (PAC) learning, which is also called Vapnik-Chervonenkis (VC) theory. This section is a brief review of VC theory and is independent of the rest of the paper. Other brief reviews are by Abu-Mostafa (1989) and Hertz *et al.* (1991); more detailed introductions are given by Vapnik (1982) and Parrando and Van den Broeck (1992).

VC theory stems from the anxiety that although a network may perform satisfactorily on the training set, it may perhaps perform differently on all possible questions; i.e., $(1/p)E_t(\mathcal{N})$ (which is observable, since it is just the behavior of our student on the training set) may not be similar to $\epsilon_f(\mathcal{N},V)$ (which is not observable, since V is unknown). How large must the training set be before these two are close?

Using advanced probability theory and assuming that questions are always drawn from some constant distribution, Vapnik and Chervonenkis (1971) were able to prove that for *any* network (in a student space such that the rule is learnable) the probability that the discrepancy between the performance on the training set and the performance on new questions will be greater than a value $\wp$ falls off very quickly with $\wp$. More formally,

$$P\left[\max_{\mathcal{N}}\left|\frac{1}{p}E_t(\mathcal{N})-\epsilon_f(\mathcal{N},V)\right|>\wp\right]\leq 4m(2p)e^{-\wp^2 p/8}\ , \tag{2.23}$$

for any V.

The function m is a measure of the size of the student space. Technically $m(p)$ is defined as the total number of different Boolean functions which can be performed by members of the student space on the p examples in the training set. For example, if the student space were so large that it contained all Boolean functions, then $m(p)$ would be the number of different Boolean functions which can be performed on p examples, that is, $m=2^p$.

Remarkably, it has been shown (Vapnik and Chervonenkis, 1971) that, for any student space, $m(p)$ is 2^p for p less than some constant d_{VC}. For $p>d_{\mathrm{VC}}$, it is bounded from above by $(ep/d_{\mathrm{VC}})^{d_{\mathrm{VC}}}$, where e is the base of natural logarithms. In the second case, Eq. (2.23) tends to zero exponentially quickly as $p\to\infty$, so that all networks in the student space must soon behave in the same way on the training set as they would on all possible questions.

In particular, this means that if we generate a network $\mathcal{N}$ which makes $E_t(\mathcal{N})$ zero, then its generalization error ϵ_g is bounded so that

$$\begin{aligned}P(\epsilon_g>\wp)&\leq 4m(2p)e^{-\wp^2 p/8}\\&\leq 4\exp[d_{\mathrm{VC}}\ln(ep/d_{\mathrm{VC}})-\wp^2 p/8]\ ,\end{aligned} \tag{2.24}$$

Thus we have a guarantee that a network which learns the training set will be able to generalize with an error less than $\wp$ for any p and $\wp$ which make the right-hand side of Eq. (2.24) small. For example, to have a 98% confidence in generalizing with an error of less than 1%, we must find a network in the student space which learns p patterns given by

$$\tfrac{2}{100}\leq 4\exp\{d_{\mathrm{VC}}\ln(ep/d_{\mathrm{VC}})-p/[8(100)^2]\}\ . \tag{2.25}$$

Clearly, d_{VC} is a critical measure of the size of student space, and it is called the Vapnik-Chervonenkis dimension of the space. Roughly, it measures the number of degrees of freedom in a student space. For a spherical binary perceptron (Sec. II.A.2), for example, the VC dimension is just N (this is demonstrated, in passing, in Sec. VI.B.1).

We may now make a few comparisons between the VC approach and the statistical-mechanics one.

Statistical mechanics, as in Sec. II.D, often makes exact predictions for the success of learning schemes. VC theory merely places bounds on their success, and the bounds are often weak. Contrariwise, it may be argued that the VC bound is rigorous, while the statistical result is only extremely *likely* to be true.

It has been found that there are many more intelligent strategies for learning than the minimization of E_t (cf. Sec. III.B), so that the agreement between E_t/p and ϵ_f is not the only important quantity. Other learning strategies fit very naturally into a statistical-mechanics formulation, but it is not immediately clear that the VC approach may be similarly extended.

VC theory, unlike statistical mechanics, gives very little insight into the student space, and for example, does not tell us how easy networks with zero E_t are to generate. We shall see that just this sort of information may be extracted from statistical mechanics.

A great strength of VC theory is that it applies to any network and indeed to any computational task, in contrast to statistical mechanics, which must be solved individually for any given network architecture. Unfortunately, the VC dimension of multilayer networks is very hard to calculate, and so far only bounds on the VC dimension are known (Baum and Haussler, 1989). Recently attempts have been made to estimate d_{VC} by experiments (Levin *et al.*, 1992; Vapnik, 1992), but it is not yet entirely clear whether this strategy has been successful.

In summary, the statistical-mechanics formulation of learning a rule may be applied, usually exactly, to all those problems for which a VC solution is available. The predictions made by the statistical-mechanics approach are more reliable in an average sense (they are overwhelmingly likely to be true) and also more informative. Of course, the real test of the usefulness of either theory is how useful it is in practice, and this is still unknown. Both techniques are presently under investigation; so their relative merit remains an open question. As we shall argue in Sec. VII.C, an important direction of future research is to reconcile them.

F. The Bayes algorithm

In this section we develop an *information-theoretic* approach to learning using Bayesian probability theory. This allows us to define the optimal way of generalizing from examples. The formalism is developed further in

Sec. II.G, where we shall be able to define the optimal way of learning a rule with a network.

1. Bayesian probability

Probability may be framed in two self-consistent but inequivalent ways. Firstly, and most commonly, the probability of an event A occurring given a set of circumstances B, which we write as $P(A/B)$, may be defined as the relative frequency of occurrences of A after B in a large number of trials. For example, an unbiased coin has probability $\frac{1}{2}$ of coming up heads (outcome A) if it is tossed (situation B). In Sec. II.B we introduced $P(V)$ in just this way: V is a single rule drawn from a hypothetical distribution $P(V)$.

An alternative Bayesian definition of probabilities is that they represent our *knowledge* or *degree of belief* in hypotheses (this idea is discussed in much greater detail by Jeffreys, 1939, and Jaynes, 1983). For any hypothesis A and any prior knowledge I, a numerical value between 0 and 1 represents our confidence that A is true. For example, if an unfamiliar coin is tossed, we have no knowledge of whether it will fall as "heads" or "tails." By symmetry there is "half a chance" that the coin will come up heads, because even if it is biased there is an "equal chance" that it is biased towards heads or tails. Thus the Bayesian probability that the coin will fall as heads next time (hypothesis A) given only that it is being tossed (information I) is $P(A/I)=\frac{1}{2}$, which may be quite different from the unknown "relative frequency of heads."

A self-consistent system of "probabilities" can be constructed in Bayesian terms with no reference to relative frequencies. Bayesian probabilities combine according to the same algebraic rules as frequentist probabilities [in fact, frequentist probabilities may be seen as a special case of Bayesian ones (Jaynes 1983)].

Bayesian probabilities can be updated by new evidence. Suppose that we have k hypotheses, labeled by a variable j, $\{A_j\}$, $j=1,\ldots,k$. Label our initial knowledge again by I, which gives us, just as in the case of the coin, a degree of belief in the truth of each proposition of $P(A_j/I)$ (which is called the *prior probability* of A_j). Note that the total belief in all hypotheses must add up to 1: $\sum_{j=1}^{k}P(A_j/I)=1$.

What is the effect on our beliefs of new data D? Suppose that hypothesis A_j would produce result D with probability $P(D/A_j,I)$; then Bayes theorem is that the degree of belief we should have in A_j after data D has arrived (the *posterior probability* of A_j) is

$$P^{\text{post}}(A_j/D,I)=\frac{P(D/A_j,I)P(A_j/I)}{\sum_{j=1}^{k}P(D/A_j,I)P(A_j/I)}, \quad (2.26)$$

where the denominator is present to ensure that the sum of the posterior probabilities remains normalized: $\sum_{j=1}^{k}P(A_j/D,I)=1$.

Example: A bag contains two balls, each of which may be red or white. A ball is removed, found to be red, and replaced. What is the probability that the bag contains two red balls?

We can define three hypotheses A_1, A_2, and A_3. A_1 is that the bag contains no red balls, A_2 is that it contains one, and A_3 is that both balls are red. We are told that each ball is either red or white (prior information I), so that it is easy to see that $P(A_1/I)=P(A_3/I)=\frac{1}{4}$ while $P(A_2/I)=\frac{1}{2}$. The data in this case are the observation that one randomly chosen ball is red (data D). The chances of this happening if one of the hypotheses is true are $P(D/A_1,I)=0$, $P(D/A_2,I)=\frac{1}{2}$, and $P(D/A_3,I)=1$. Thus, using Bayes theorem [Eq. (2.26)], we obtain the posterior probability $P(A_3/D,I)=\frac{1}{2}$, which is the answer to the problem.

Bayesian ideas have made a substantial contribution to physics, particularly in the formulation of statistical physics without reference to the ergodic hypothesis (Jaynes, 1957, 1983).

In learning theory, the hypotheses are the rule we believe has generated the data. We take our prior information I as the knowledge that a rule is a member of a given rule space. For example, we might know that the rule is linearly separable [i.e., that it can be written as Eq. (2.10)]; by symmetry, the $\mathbf{B}$ that defines the rule is equally likely (without further knowledge) to be anywhere on the sphere. Thus the prior probability of a rule, $P(\mathbf{B}/I)$, is uniform. Henceforth we shall assume that we are working in a given rule space and for simplicity omit the I from our equations. The setting of $P(V)$ in more advanced and realistic rule spaces is discussed further in Sec. VII.A.

Let us now consider the meaning of "expectation values" under the frequentist and Bayesian definitions of probability.

Under the frequentist formulation, expectation values of numerical quantities are exact predictions of the mean observation in a very large number of trials—for example, the mean number of spots shown in many spins of an unbiased die *will* tend to 3.5, as the number of trials tends to infinity. Thus working out averages over $P(V)$, as in Eq. (2.14), means finding the mean behavior in learning many rules taken from $P(V)$. In practice, this is not really a problem because almost all realizations of V taken from a given $P(V)$ turn out to give the same values of interesting phenomena, which are said to be *self-averaging* (as argued in Sec. III.A.2). That is, fluctuations are small; so phenomena found while learning any single rule closely resemble the mean phenomena.

Under the Bayesian formulation, the "expectation value" of a quantity is the sum over hypotheses of the value of the quantity if a hypothesis is true weighted by our belief in the hypothesis. We shall see in learning that such sums are dominated by hypotheses which give the same value for the interesting quantities; i.e., fluctuations are small. This means that the knowledge we have leads

us to make very firm predictions about observations in a single experiment.

In summary, the techniques we develop in this paper may be formulated in conventional frequentist terms or in Bayesian terms. In either case important quantities will be firmly predicted by expectation values. However, we shall show in the next sections how Bayesian ideas can be used to extract extra information about generalization.

2. The principle of the Bayes algorithm

There is an *optimal,* information-theoretic prescription for predicting the answer to a new question from known examples. Using Bayes theorem, Eq. (2.26), we can calculate the posterior probability of any given rule having generated the training set. The data D in this case is the training set $\{\xi^\mu,\xi_o^\mu\}$. If the data are noiseless, then $P(D/V)=\prod_\mu \delta(V(\xi^\mu),\xi_o^\mu)$, where we are using the Kronecker delta $\delta(a,b)$, which is 1 if $a=b$ and zero otherwise. Then, using Bayes theorem, the posterior probability of a rule is

$$P^{\text{post}}(V/D)=\frac{P(D/V)P(V)}{\int dV\,P(D/V)P(V)} . \tag{2.27}$$

Suppose we have a new question $\mathbf{r}$ to answer. How should we do it? Clearly the answer depends upon our belief in a hypothesis such as "the true answer to $\mathbf{r}$ is $\mathcal{R}$," which for simplicity we call hypothesis $\mathcal{R}$. Clearly, too, our belief in this hypothesis is the sum of our belief in all the rules which would answer $\mathbf{r}$ by $\mathcal{R}$. That is,

$$P^{\text{post}}(\mathcal{R}/D)=\int dV\,P^{\text{post}}(V/D)\delta(V(\mathbf{r}),\mathcal{R}) . \tag{2.28}$$

Substituting Eq. (2.27) into Eq. (2.28) gives

$$P^{\text{post}}(\mathcal{R})=\frac{\int_{\mathcal{V}(p)}dV\,P(V)\delta(V(\mathbf{r}),\mathcal{R})}{\int_{\mathcal{V}(p)}dV\,P(V)} . \tag{2.29}$$

This is the proportion of the version space (weighted by the prior distribution of V) which would give the answer $\mathcal{R}$, that is, the chance that the answer is $\mathcal{R}$.

The information-theoretic (Bayesian) prediction of the answer to the new question $\mathbf{r}$ is the $\mathcal{R}$ which has the maximum posterior probability. The chance that the Bayes algorithm answers $\mathbf{r}$ wrongly, $\epsilon^{\text{Bayes}}(\mathbf{r})$, is the chance that the true V lies in one of the regions of version space with a different $\mathcal{R}$. The average Bayesian generalization error, $\epsilon_g^{\text{Bayes}}$, is the average of $\epsilon^{\text{Bayes}}(\mathbf{r})$ over the distribution of $\mathbf{r}$.

3. Statistical mechanics of the Bayes algorithm—an example

This example builds on the situation of a perceptron learning a linearly separable rule introduced in Sec. II.B.2. The first two examples on Fig. 4(d) had reduced the version space to the region between $\mathcal{D}_1$ and $\mathcal{D}_2$.

If the new question asked is $\mathbf{r}$, which is marked on Fig. 6(a) perpendicular to plane $\mathcal{D}_\mathbf{r}$, and the prior distribution

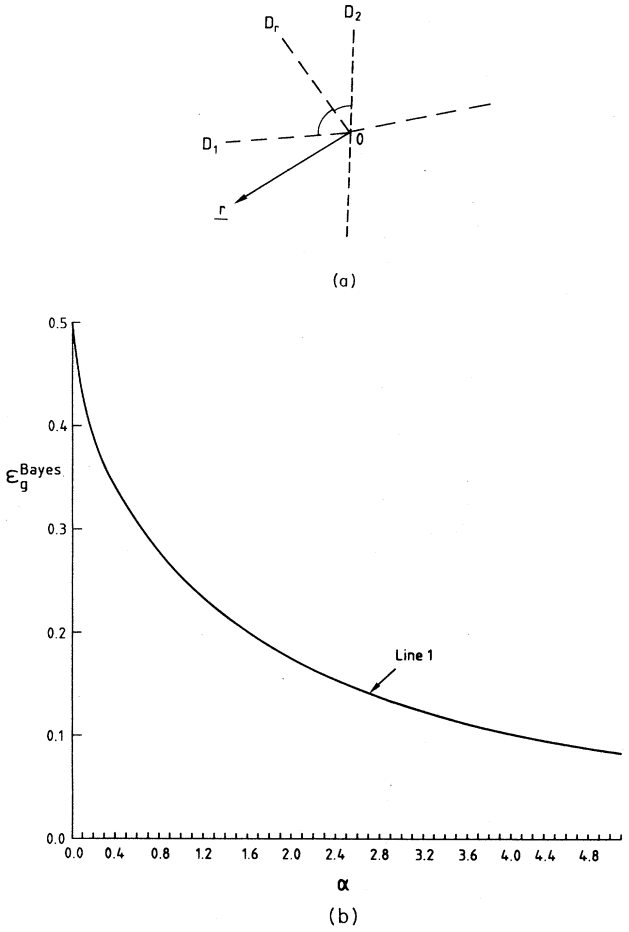

FIG. 6. Bayes algorithm for a linearly separable rule: (a) question space, with a new question $\mathbf{r}$ added, perpendicular to hyperplane $\mathcal{D}_\mathbf{r}$, (b) optimal, Bayesian generalization error, $\epsilon_g^{\text{Bayes}}$ against α.

of $\mathbf{B}$ is uniform, the posterior probability of the answer being $+1$ is the proportion of the version space between $\mathcal{D}_1$ and $\mathcal{D}_\mathbf{r}$, since any $\mathbf{B}$ vector in this region answers $\mathbf{r}$ by $+1$. Similarly, the posterior probability for -1 is the proportion of the version space between $\mathcal{D}_\mathbf{r}$ and $\mathcal{D}_2$. Notice that there are some choices for question $\mathbf{r}$ which make one of these posterior probabilities equal to zero. In Fig. 6(a) it appears that the posterior probability of $+1$ is the larger; so the Bayes algorithm predicts $\mathcal{R}=1$.

Here $\epsilon^{\text{Bayes}}(\mathbf{r})$ is the proportion of version space which would answer $\mathbf{r}$ by $\mathcal{R}=-1$; that is, $\epsilon^{\text{Bayes}}(\mathbf{r})=P^{\text{post}}(\mathcal{R}=-1)$. The variable $\epsilon_g^{\text{Bayes}}$ is the averaged chance that the true $\mathbf{B}$ belongs to the part of the space which gives the opposite answer to the Bayes algorithm,

$$\epsilon_g^{\text{Bayes}}\equiv\int d\mu(\mathbf{r})\epsilon^{\text{Bayes}}(\mathbf{r}) , \tag{2.30}$$

where $d\mu(\mathbf{r})$, as in Eq. (2.12), denotes the measure of the distribution of questions.

Assuming only that the overlaps between the questions and $\mathbf{B}$ are distributed with zero mean overlap and in such

a way that the central limit theorem (see, for example, Bronstein and Semendjajew, 1985) can be applied, which would be the case, for example, if every component of $\mathbf{B}$ and of every question were randomly ± 1, and that the questions are independent, Opper and Haussler (1991a) have shown (using the method of replicas, which is explained in its more usual context in Sec. III) that asymptotically, as $N \to \infty$,

$$\epsilon_g^{\text{Bayes}} = \frac{1}{\pi} \cos^{-1}(\sqrt{q}) \ . \tag{2.31}$$

Here q is the average overlap $\mathbf{B}^\gamma \cdot \mathbf{B}^\delta$ of two rules $\mathbf{B}^\gamma$ and $\mathbf{B}^\delta$, chosen randomly in the version space in proportion to the prior probability $P(\mathbf{B})$. If, in addition, the prior distribution of rules is uniform, the value of q has been calculated as a function of p (Györgyi and Tishby, 1990). We can therefore plot $\epsilon_g^{\text{Bayes}}$ against α as line 1 in Fig. 6(b); for large α, $\epsilon_g \sim 0.44/\alpha$.

Although this sort of calculation gives the information-theoretic upper limit for the generalization ability, it should be emphasized that it does not itself correspond to a realistic learning process. The V-space is usually a very large one, and the boundaries of $\mathcal{V}^{(p)}$ formed by random examples are too complicated for an exact calculation of posterior probabilities to be reasonable.

G. Optimal learning

1. Bayesian reformulation of learning

Ideally, we would like the network we build to implement the Bayes algorithm described in Sec. II.F since this minimizes the average generalization error. Such a network, however, would require a considerably more complex structure than the rule, since it would need to be capable not just of learning the rule V, but also of encoding the whole structure of version space (defined above as all the rules consistent with the examples we have been given). Indeed, as we shall see in Sec. VI.C, even implementing the Bayes algorithm exactly for a linearly separable rule requires a two-layer *committee machine* with a number of neurons in the hidden layer rising to infinity. In practice the rule is likely to be *more* complex than the student network we are allowed to build. That is, the student space may not even have a member which exactly learns V. How, then, should $\mathcal{N}$ be chosen?

To answer this question, we must take a Bayesian view of the formalism of learning. In Sec. II.F.2 we were able to calculate the posterior probability of any rule, Eq. (2.27), while the generalization function, Eq. (2.12), is a measure of the "badness" of any network $\mathcal{N}$ given the true rule V. By combining these results, we are able to define a measure of the badness of any network $\mathcal{N}$ in which our ignorance of the underlying rule has been integrated out. The result is called the *network error,*

$$\epsilon_n(\mathcal{N}) = \int dV \, \epsilon_f(\mathcal{N}, V) P^{\text{post}}(V/D) \ . \tag{2.32}$$

In the case of noiseless examples, so that the posterior probability of rules is only nonzero in the version space, this reduces to

$$\epsilon_n(\mathcal{N}) = \langle \epsilon_f(\mathcal{N}, V) \rangle_{\mathcal{V}} \ , \tag{2.33}$$

where $\langle \cdots \rangle_{\mathcal{V}}$ simply indicates the average over version space. In either case, $\epsilon_n(\mathcal{N})$ is the expectation generalization error in answering new questions (Watkin, 1991, 1993).

We may thus construct a Bayesian form of the average generalization error of networks produced by an algorithm,

$$\begin{aligned} \epsilon_g(\{\xi^\mu, \xi_o^\mu\}) &\equiv \langle \epsilon_n(\mathcal{N}) \rangle_{\mathcal{N}} \\ &= \left\langle \left\langle \int d\mu(\mathbf{S}) e(\mathcal{N}, V, \mathbf{S}) \right\rangle_{\mathcal{V}} \right\rangle_{\mathcal{N}} \ . \end{aligned} \tag{2.34}$$

For a given realization of the training set, ϵ_g is the expectation error made by a network in answering new questions.

Equation (2.34) is observable, since the dependence upon unknown variables has been integrated out. $\epsilon_g(\{\xi^\mu, \xi_o^\mu\})$ is a function only of the training set. This is different from $\epsilon_g(V, \{\xi^\mu\})$, defined by Eq. (2.13), which is a function of the unknown rule.

$\epsilon_g(\{\xi^\mu, \xi_o^\mu\})$ determines how well an algorithm generalizes for a given training set. But we can also work out the *a priori* likelihood of a given training set being generated. This means that we can calculate from first principles the expectation values of how well we *would* be able to generalize, *if* we were given p examples of the rule.

For a given realization of the set of questions $\{\xi^\mu\}$ we can decompose the rule space into regions which answer the questions in the same way. Let us, in this section, label one possible set of answers by $a \equiv \{\xi_o^\mu\}$. The "likelihood" of this set of answers occurring is the proportion of rule space giving this set of answers, $P(a/\{\xi^\mu\}) = \int dV \, P(V) \prod_\mu \delta(V(\xi^\mu), \xi_o^\mu)$. The average over the whole rule space is the weighted sum of the averages over these regions; thus the expectation value of our ability to generalize from p examples is

$$\begin{aligned} \langle \langle \epsilon_g(\{\xi^\mu, \xi_o^\mu\}) \rangle_a \rangle_\xi &= \left\langle \sum_a P(a/\{\xi^\mu\}) \langle \langle \epsilon_f(\mathcal{N}, V) \rangle_{\mathcal{N}} \rangle_{\mathcal{V}} \right\rangle_\xi \\ &= \left\langle \sum_a P(a/\{\xi^\mu\}) \int dV \, P^{\text{post}}(V/\{\xi^\mu\}, a) \langle \epsilon_f(\mathcal{N}, V) \rangle_{\mathcal{N}} \right\rangle_\xi \ , \end{aligned}$$

where, once again, $\langle \cdots \rangle_\xi$ denotes the average over the set of questions. But substituting $P^{\text{post}}(V/\{\xi^\mu\}, a)$ from Eq.

(2.27) results in the denominator of P^{post} canceling with $P(a/\{\xi^\mu\})$, leaving

$$\langle\langle\epsilon_g(\{\xi^\mu,\xi_o^\mu\})_a\rangle_\xi=\left\langle\int dV\left[\sum_a P(a/V,q)\right]P(V)\langle\epsilon_f(\mathcal{N},V)\rangle_{\mathcal{N}}\right\rangle_\xi . \tag{2.35}$$

The term in the round brackets is just 1; so

$$\langle\langle\epsilon_g(\{\xi^\mu,\xi_o^\mu\})_a\rangle_\xi=\langle\langle\langle\epsilon_f(\mathcal{N},V)\rangle_{\mathcal{N}}\rangle_V\rangle_\xi , \tag{2.36}$$

which is clearly the same as Eq. (2.14).

Thus after the average over quenched disorder has been carried out, the generalization error defined by these two procedures can meaningfully be compared. In fact, we can exploit the self-averaging nature of the situation and say that almost all training sets of the same size give an equal ability to learn. Thus in Sec. III.A.3 we shall frame statistical mechanics as calculating typical properties of a version space. This will let us easily find values for how well an algorithm would learn from p examples.

2. Optimal learning

Watkin (1991, 1993) pointed out that for a given training set the optimal learning algorithm is to construct $\mathcal{N}$, which minimizes $\epsilon_n(\mathcal{N})$ within the student space, since, from Eq. (2.34), this algorithm would minimize the expectation values of generalization error, $\epsilon_g(\{\xi^\mu,\xi_o^\mu\})$. This strategy is optimal learning for any network learning any rule from any data, since by definition it minimizes the expectation generalization error.

How are we to implement it? In principle we could try to minimize an extensive energy

$$E_{\text{opt}}(\mathcal{N})\equiv p\epsilon_n(\mathcal{N})=p\langle\epsilon_f(\mathcal{N},V)\rangle_{\mathcal{V}} , \tag{2.37}$$

which is observable, since the dependence on the unknown V has been integrated out by the average over $\mathcal{V}$; but this is a very difficult quantity to observe: we must calculate the whole complex structure of the version space to evaluate it. However, Watkin (1991, 1993) generalized a method first devised by Opper and Haussler (1991a) to solve a different problem (which is described in Sec. VI.C.1) and sketched an algorithm to find $E_{\text{opt}}(\mathcal{N})$ which requires only *sampling* of the version space and which, in principle, would work for any network learning any rule. The optimal network may then be found either by minimizing $E_{\text{opt}}(\mathcal{N})$ or, in simple cases, by construction. We give an example of this algorithm in the context of perceptron learning in Sec. III.B.2.

3. An amusing paradox

An amusing paradox, which also illustrates some of the power of the tools developed above, was pointed out by Seung (private communication, 1991). Suppose that student space contains only three elements $\mathcal{N}_1$, $\mathcal{N}_2$, and $\mathcal{N}_3$, and that the whole space of questions has Q elements labeled by $q=1,\ldots,Q$. Let the answer that $\mathcal{N}_1$ makes to the qth question be a_q and suppose that every question has only two possible answers, the alternative being b_q. Thus the answers $\mathcal{N}_1$ gives to the set of questions are $(a_1,a_2,\ldots,a_Q)$. Let $\mathcal{N}_2$ give answers $(a_1,b_2,b_3,\ldots,b_Q)$, and $\mathcal{N}_3$ give answers $(b_1,a_2,a_3,\ldots,a_Q)$. Now let us suppose that the true rule we are learning, V, happens to agree with $\mathcal{N}_1$ on all examples. Then $\epsilon_f(\mathcal{N}_1,V)=0$, $\epsilon_f(\mathcal{N}_2,V)=1-(1/Q)$, and $\epsilon_f(\mathcal{N}_3,V)=1/Q$, which in each case is the proportion of questions each network gets wrong. Since we have not so far been given any examples, $\mathcal{N}_1$, $\mathcal{N}_2$, and $\mathcal{N}_3$ are the three degenerate minima of E_t and are equally likely to be generated. Since on all questions two of the three get the answer to each question right, the average error is $\frac{1}{3}$, runs the argument.

We are now presented with the correct answer a_1 to question 1, which $\mathcal{N}_3$, of course, gets wrong. Thus $\mathcal{N}_1$ and $\mathcal{N}_2$ are two degenerate minima of E_t and are equally likely to be generated by an algorithm which just minimizes E_t. They disagree, however, on all the $(Q-1)$ other questions; so the average error is $(Q-1)/(2Q)$, which is approximately $\frac{1}{2}$ for Q large. This is larger than $\frac{1}{3}$, the argument concludes, so would it not have been better to ignore the example altogether? In Sec. III.A.5 we shall call this using a high training temperature.

The fallacy in the paradox is exposed by looking more closely at the formalism above. Since we are not given the underlying rule, the only reasonable strategy is to minimize $\epsilon_g(\{\xi^\mu,\xi_o^\mu\})$, which contains an average over the version space: it is not correct to arbitrarily take V as agreeing with $\mathcal{N}_1$. Given no information, the prior probability of a rule gives equal weight to all Boolean functions on the Q questions. All three networks therefore have a network error of $\epsilon_n=\frac{1}{2}$, and if each is equally likely to be produced, $\epsilon_g=\frac{1}{2}$. Given, in addition, the answer to the first question, the remaining version space is all Boolean functions which answer the first question by a_1; since the two degenerate minima of E_t, $\mathcal{N}_1$ and $\mathcal{N}_2$, disagree on the answer to all the $(Q-1)$ other questions, the ϵ_g produced by choosing a network to minimize E_t is $(Q-1)/(2Q)$, which is lower than $\frac{1}{2}$. Thus the paradox is resolved. Ignoring the first example, so that $\mathcal{N}_1$, $\mathcal{N}_2$, and $\mathcal{N}_3$ are equally likely to be generated, would mean that the net would also have $\frac{1}{3}$ of a chance of answering question 1 incorrectly.

Of course, if we had prior information that $\mathcal{N}_1$ learned V perfectly, then optimal learning would *not* be to minimize E_t (if we knew the answer, why are we learning at all?). A case is presented in Sec. V.A.4 in which prior information about the rule being learned means that optimal learning generates an $\mathcal{N}$ which is not a minimum of

E_t; so it may indeed be better to use a nonzero training temperature.

III. LEARNING AS A STOCHASTIC PROCESS

Here we shall show how statistical mechanics has been applied to the problem stated in Sec. II of calculating average generalization errors, and how closely the consequences are related to the phenomena of disordered physics. To give the reader a flavor of a typical calculation, we summarize the general formalism and develop in parallel the example of the binary spherical perceptron, which was introduced in Sec. II.

A. The general formalism and its application to the perceptron

1. Introduction

In the previous sections we have introduced a number of cost functions and given the suggestive name of "energies." The simplest is the training energy $E_t(\mathcal{N})$, which measures the number of known examples of the underlying rule which a network $\mathcal{N}$ gets wrong, Eq. (2.11). Since we want a network to be able to generalize properly, an obvious strategy is to build one which at least gets the known examples of the rule right, that is, $\mathcal{N}$ for which $E_t(\mathcal{N})=0$. This strategy is very widespread in engineering practice (see Rumelhart and McClelland, 1986, for reviews).

The two methods which are most commonly used to construct such an $\mathcal{N}$ are *backpropagation* (Sec. VI.B.3), which is essentially gradient descent on the landscape given by $E_t(\mathcal{N})$, and *constructive algorithms,* in which new nodes are added to a network until it is large enough to learn the training set easily (Sec. VI.E). Both methods have disadvantages. Backpropagation, like other gradient descent algorithms, is liable to become trapped in local minima of the energy surface and so rarely finds a network with zero training energy. Constructive algorithms, on the other hand, do make $E_t(\mathcal{N})=0$, but often produce a network which is very much more "complex" than the rule being learned, which leads to poor generalization. Further insight into learning is required for making systematic progress in either case.

It turns out that backpropagation and constructive algorithms are both difficult to analyze directly. A solution is to study how a stochastic algorithm would behave on the landscape defined by E_t, an idea formulated by Carnevali and Patarnello (1987), Patarnello and Carnevali (1987), Denker *et al.* (1987), and Levin *et al.* (1989), in the hope of understanding the landscape itself. Exact statistical mechanics to calculate the success of stochastic algorithms was then carried out by Tishby *et al.* (1989), Gardner and Derrida (1989), Hansel and Sompolinsky (1990), Györgyi (1990a), Györgyi and Tishby (1990), Opper *et al.* (1990), and Sompolinsky *et al.* (1990). This will be discussed in Sec. III.A.2.

A second justification for introducing statistical mechanics is discussed in Sec. III.A.3: to work out typical properties of the version space (Opper and Haussler, 1991a; Watkin, 1993). Either justification leads to the same mathematical problems, which can be resolved using the method of replicas, familiar from spin-glass theory.

Section III is the only one to contain any algebra. We have included it to give the reader some flavor of research in the field. All other sections describe conclusions which can either be deduced from the results of this calculation or which are straightforward variations.

2. Dynamic approach

The approach which is clearest for most physicists is one in which the algorithm which generates networks resembles the dynamics of physical systems evolving according to an energy. For simplicity we shall specialize here to networks which, like the perceptron, may be described by a single vector $\mathbf{J}$, so that, for example, we can write the training energy as $E_t(\mathbf{J})$.

The space of couplings can be explored by considering a stochastic learning process using *any* observable energy E. This may be the extensive energy $E_t(\mathcal{N})$ defined in Sec. II.C, or $E_{\rm opt}$ introduced in Sec. II.G.2, or another energy, $E_{\rm MSA}$, which will be defined in Sec. III.B.1. For continuous $\mathbf{J}$ and a differentiable energy, we allow the couplings to evolve according to a Langevin dynamics (Langevin, 1908; Itzykson and Drouffe, 1989), which has the same form as the dynamics of a physical system in the limit of *high viscosity* (Parisi, 1988), or on a *velocity-dependent* energy surface.

$$\frac{\partial \mathbf{J}}{\partial t} = -\nabla_{\mathbf{J}} E(\mathbf{J}) + \mathbf{F}(t) \ , \tag{3.1}$$

where $\mathbf{F}(t)$ is white noise, with the property $\langle F_i(t)F_j(t')\rangle = 2T\delta_{ij}\delta(t-t')$, and T is the effective temperature (here, and throughout this paper, the units are chosen so that Boltzmann's constant is 1). Seung, Sompolinsky, and Tishby (1992) pointed out that possible constraints on the $\mathbf{J}$-space can be enforced in Eq. (3.1) by including a component in the energy which does not depend upon the examples; henceforth we shall assume that this has been carried out so as to enforce the gauge constraint we are imposing on the system, which for a spherical perceptron is Eq. (2.8).

For Ising $\mathbf{J}$ or for a nondifferentiable energy function, one can use a Metropolis Monte Carlo spin-flip dynamics (Köhler *et al.*, 1990; Horner, 1992a, 1992b), a method reviewed by Binder and Heerman (1988).

Either stochastic dynamics—when certain specific constraints are fulfilled—generates a Gibbs distribution in $\mathbf{J}$-space, that is, couplings whose probability of occurrence, *measure,* is

$$d\mu_G(\mathbf{J}) \equiv \frac{d\mu(\mathbf{J})}{Z} e^{-\beta E(\mathbf{J})}, \quad \beta \equiv \frac{1}{T} \ , \tag{3.2}$$

where $d\mu(\mathbf{J})$ is the normalized *a priori* measure on the J-space, and Z is the partition function

$$Z \equiv \int d\mu(\mathbf{J})\, e^{-\beta E(\mathbf{J})} . \tag{3.3}$$

Thus for a J which is equally likely to be in any direction and whose magnitude is simply fixed by Eq. (2.8), the measure on the weight space is just $d\mu(\mathbf{J}) \equiv d\mathbf{J}\,\delta(\mathbf{J}\cdot\mathbf{J}-1)$. Similarly for Ising weights, if any combination of weights is equally likely, it is $d\mu(\mathbf{J}) \equiv d\mathbf{J}\prod_i [\frac{1}{2}\delta(J_i-1)+\frac{1}{2}\delta(J_i+1)]$. The inverse temperature β, introduced in Eq. (3.2), determines the extent to which J is allowed to explore the weight space: $\beta\to\infty$ forces J into the minimum of E; $\beta\to 0$ opens up the whole phase space. In general, β determines the energy we tolerate. A nonzero training temperature is particularly important when trying to learn unlearnable rules (Sec. V.A). Note that if the energy E used in this formulation is the training energy E_t, then, in the limit of $\beta\to\infty$, Z is the fraction of the student space with $E_t(\mathcal{N})=0$.

Based on this framework we can now use the tools of statistical mechanics to calculate the corresponding thermal averages. For example,

$$\langle E(\mathcal{N})\rangle_{\mathcal{N}} = -\frac{d}{d\beta}\ln Z . \tag{3.4}$$

We could also introduce an additional, auxiliary term into the energy:

$$E \to E + h\epsilon_f(\mathcal{N}, V) , \tag{3.5}$$

where the auxiliary field h will later be set to zero. Then, for a given underlying realization of V and the set of questions,

$$-\frac{1}{\beta}\frac{d}{dh}\ln Z\bigg|_{h=0} = \langle \epsilon_f(\mathcal{N}, V)\rangle_{\mathcal{N}} . \tag{3.6}$$

Obviously, we could work out the expectation value of any quantity by changing the coefficient of h.

Inserting Eq. (3.6) into Eq. (2.14) shows that the interesting quantities are obtained by calculating derivatives of

$$F \equiv -\frac{1}{\beta}\langle\langle \ln Z\rangle_V\rangle_\xi . \tag{3.7}$$

Why should we expect such quantities to be self-averaging?

Energy landscapes which depend upon the realizations of random variables (in our case the training set) are familiar in many other fields of physics. A well-known example is the field of spin glasses, spin systems in which every pair of spins has a random ferromagnetic or antiferromagnetic interaction [an enjoyable and remarkably concise introduction to the theory is given by the series of articles by Anderson (1988–1990)].

In these systems it is usually assumed that values of extensive quantities do not depend critically upon the realization of the quenched variables—almost all realizations of quenched disorder generated by the same microscopic distribution lead to the same macroscopic quantities, which are therefore said to be self-averaging. For a given realization of the quenched variables, one extensive quantity is the free energy, $-(1/\beta)\ln Z$, whose derivatives give us observable quantities. If the free energy is self-averaging, then almost all realizations of the quenched variables give a free energy equal to the average free energy $F \equiv [-(1/\beta)\ln Z]_{\rm av}$, where the square bracket indicates the average over quenched disorder; and so we can make correct predictions of real observations using F.

We shall expect, and also be able to show (assuming the validity of the replica method), that most interesting, macroscopic quantities *are* self-averaging. For example, it will be possible to show for a spherical perceptron learning a linearly separable rule, that

$$[\epsilon_g]_{\rm av} = \frac{1}{\pi}\cos^{-1}([\langle \mathbf{B}\cdot\mathbf{J}\rangle_{\mathcal{N}}]_{\rm av}) . \tag{3.8}$$

Later in Sec. III we shall calculate $[\langle \mathbf{B}\cdot\mathbf{J}\rangle_{\mathcal{N}}]_{\rm av}$ for a number of algorithms which construct J from a training set; we shall therefore have calculated the average generalization error.

3. Bayesian approach

The previous section provided one description of the place of statistical mechanics in neural network theory: in determining how a stochastic algorithm generates a network. Here we shall discuss an alternative approach: using statistical mechanics to tell us about the typical behavior of version space, without reference to a network trying to learn the rule.

The posterior probability of a rule, Eq. (2.27), can be rewritten as

$$P^{\rm post}(V/\{\xi^\mu,\xi_o^\mu\}) = \lim_{\beta\to\infty}\frac{e^{-\beta E_t(V,\{\xi^\mu,\xi_o^\mu\})}P(V)}{Z} , \tag{3.9}$$

where E_t is the training energy, defined by Eq. (2.11), and

$$Z = \int dV\, e^{-\beta E_t(V,\{\xi^\mu,\xi_o^\mu\})}P(V) . \tag{3.10}$$

Using Eq. (3.9) we could construct many different measures of our knowledge about the underlying rule. For example, we could work out the distribution of the overlap of two points in rule space chosen randomly according to Eq. (3.9). Just as in the stochastic approach, the numerical value of any such measure could be found from derivatives of $\ln Z$.

These derivatives tell us the information we would have about V *if* we had the training set $\{\xi^\mu,\xi_o^\mu\}$. Thus the expectation values of properties of the version space, *if* we have p examples, are given by derivatives of the average of $\ln Z$ over possible training sets. A trivial argument, similar to the one which proved Eq. (2.36), shows that this is the same as the average of $\ln Z$ over $P(V)$ and over questions,

$$\langle\langle \ln Z\rangle_V\rangle_\xi . \tag{3.11}$$

Expression (3.11) is formally proportional to the average free energy of the dynamical approach, Eq. (3.7), providing that four conditions are met: (a) Rule space equals student space, so that the integral over $\mathcal{N}$-space in Eq. (3.3) is the same as that over V-space in Eq. (3.10); (b) the energy used in Eq. (3.3) is E_t; (c) $P(V)$ is uniform in the rule space; and (d) the training temperature of Eq. (3.3) is zero. We shall see in Sec. V.B that Bayesian analysis using noisy examples is formally equivalent to stochastic training at finite temperature.

Either the Bayesian or the dynamic approach (Sec. III.A.2) leads to the same mathematical problem, the evaluation of the average of a logarithm. We shall spend the rest of Sec. III.A solving this problem. Since each node in a large network has many inputs, the network is essentially *infinite* dimensional and mean-field theory will be exact (Fisher and Gaunt, 1964; Brout, 1965; Stanley, 1971). Our treatment will be within the language of the first, dynamic approach.

4. Method of replicas

The technique commonly used to perform the average over examples is the replica method, which was first devised by Kac (1968), reinvented for the analysis of rubber elasticity by Edwards (1970), and applied to spin glasses (Edwards and Anderson, 1975; Sherrington and Kirkpatrick, 1975) and neural networks (Amit *et al.*, 1985). Gardner (1987, 1988) reapplied it to the analysis of the student space of networks. It is used in cases for which performing an average of the partition function Z is easy but that of $\ln Z$ is not, and it exploits the identity

$$[\ln Z]_{\rm av}=\lim_{n\to 0}\frac{1}{n}([Z^n]_{\rm av}-1)\ . \tag{3.12}$$

The expression Z^n is equivalent to a partition function of n identical systems, replicas, labeled by $\gamma=1,\ldots,n$, which do not interact. However, performing the average over the quenched disorder couples *different* replicas. For example, if the energy E used is E_t, then

$$\langle Z^n\rangle_{V,\xi}=\int\left[\prod_\gamma d\mu(J^\gamma)\right]\left\langle\exp\left[-\beta\sum_{\mu,\gamma}e(\mathbf{J}^\gamma,\xi^\mu)\right]\right\rangle_{V,\xi}$$
$$=\int dV\left[\prod_\gamma d\mu(\mathbf{J}^\gamma)\right]P(V)e^{-p\beta\mathcal{H}(\{\mathbf{J}^\gamma\})}\ , \tag{3.13}$$

where $\mathcal{H}$, the effective Hamiltonian in the replicated space, is given by

$$\mathcal{H}(\{J^\gamma\})\equiv-\frac{1}{\beta}\ln\left[\int d\mu(\mathbf{S})\exp\left[-\beta\sum_\gamma e(\mathbf{J}^\gamma,\mathbf{S})\right]\right]\ , \tag{3.14}$$

and, as in Eq. (2.12), $d\mu(\mathbf{S})$ means the normalized distribution from which the questions $\mathbf{S}$ are chosen. This approach allows one to interpret the average of observables of systems with particular sets of questions in terms of corresponding observables of an effective replicated system, where all specific dependence on the questions has been removed.

Example: Suppose that we are analyzing the learning of a learnable rule by a binary spherical perceptron (Györgyi and Tishby, 1990; Seung, Sompolinsky, and Tishby, 1992). The natural error measure is $e(\mathbf{J},\mathbf{B},\xi^\mu)=\Theta(-N(\mathbf{B}\cdot\xi^\mu)(\mathbf{J}\cdot\xi^\mu))$, where the function $\Theta(x)$ is the usual Heaviside function: $\Theta(x)=1$, if $x>0$, and zero otherwise. Therefore, since the questions are distributed independently, we can write

$$e^{-p\beta\mathcal{H}}=\left[\left\langle\exp\left[-\beta\sum_\gamma e(\mathbf{J}^\gamma,\xi)\right]\right\rangle_\xi\right]^p\ . \tag{3.15}$$

The quantity in the square brackets is

$$\int\left[\prod_\gamma d\lambda^\gamma\right]dr\exp\left[-\beta\sum_\gamma\Theta(-\lambda^\gamma)\right]\left\langle\prod_\gamma\delta[\lambda^\gamma-\sqrt{N}\,(\mathbf{J}^\gamma\cdot\xi)\mathrm{sgn}(r)]\delta(r-\sqrt{N}\,\mathbf{B}\cdot\xi)\right\rangle_\xi\ , \tag{3.16}$$

where, as in the rest of this paper, unless otherwise stated the upper and lower limits of integrals are $\pm\infty$, respectively. We use the well-known integral representation of the δ function, so that Eq. (3.16) is

$$\int\left[\prod_\gamma\frac{d\hat\lambda^\gamma d\lambda^\gamma}{2\pi}\right]\frac{d\hat r\,dr}{2\pi}\exp\left[-\beta\sum_\gamma\Theta(-\lambda^\gamma)\right]\left\langle\exp\left[\sum_\gamma i\hat\lambda^\gamma\{\lambda^\gamma-\sqrt{N}\,[\mathbf{J}^\gamma\cdot\xi\,\mathrm{sgn}(r)]\}+i\hat r(r-\sqrt{N}\,\mathbf{B}\cdot\xi)\right]\right\rangle_\xi\ . \tag{3.17}$$

The average over ξ may now be performed straightforwardly, noting that, for any A, $\langle e^{iA\xi_i}\rangle_{\xi_i}=e^{-A^2/2}$ to first order in A^2. Then the $\hat r$ variable is integrated out to give

$$\int\left[\prod_\gamma\frac{d\hat\lambda^\gamma d\lambda^\gamma}{2\pi}\right]Dr\exp\left[-\beta\sum_\gamma\Theta(-\lambda^\gamma)+\sum_\gamma i\hat\lambda^\gamma\lambda^\gamma-i\sum_\gamma|r|\hat\lambda^\gamma(\mathbf{B}\cdot\mathbf{J}^\gamma)-\tfrac{1}{2}\sum_{\gamma,\gamma'}\hat\lambda^\gamma\hat\lambda^{\gamma'}[\mathbf{J}^\gamma\cdot\mathbf{J}^{\gamma'}-(\mathbf{B}\cdot\mathbf{J}^\gamma)(\mathbf{B}\cdot\mathbf{J}^{\gamma'})]\right]\ , \tag{3.18}$$

which contains the overlap of different replicas, implying that they have become coupled. We shall show in Sec. III.A.7 how this analysis carries on.

Before analyzing the full theory, we present two simplifying approximations.

5. High-temperature limit

If, again, E is just E_t, then the leading term in an expansion of the Hamiltonian $\mathcal{H}$, Eq. (3.14), in powers of β is the nonrandom part of the training energy, which does not couple different replicas (Sompolinsky *et al.*, 1990). Hence in the high-temperature limit (neglecting other terms) the Gibbs measure reduces to

$$d\mu_G(\mathbf{J}) \equiv \frac{d\mu(\mathbf{J})}{Z} e^{-N\alpha\beta\epsilon_f(\mathbf{J},V)} , \tag{3.19}$$

which has no dependence on the example set. This expression makes it clear that, for $\beta \to 0$, the limit is only defined in a proper way if $\alpha \to \infty$, leaving $\alpha\beta$ constant. This can be easily understood: with increasing temperature the resulting dynamical noise can only be compensated for by a number of examples scaling with the temperature. Sompolinsky *et al.* (1990) introduced a rescaled number of examples $\tilde{\alpha} \equiv \alpha\beta$. When the temperature is high, the number of examples is large; so they form a representative sample of all possible questions. The details of which examples are actually given are irrelevant, and thus the replica calculation is avoided, greatly simplifying the mathematical analysis, as is shown in Sec. III.C.1. For the same reason, in the high-temperature limit, the training error equals the generalization error.

In the high-temperature limit the dynamics effectively minimizes a free energy per weight f given by

$$\beta f = \tilde{\alpha}\epsilon_f - s , \tag{3.20}$$

where s is the entropy per weight of the system, which we shall define more carefully in Sec. III.C.1. High-T learning can be understood as a dynamical process with an effective energy $p\epsilon_f(\mathbf{J},V)$ which is *smoother* (i.e., it has fewer local minima) than $E_t(\mathbf{J})$, since it is averaged over the whole question space and not just over the training set.

Example: When Eq. (3.18) is expanded to first order in β, that is, $\exp[-\beta\sum_\gamma \Theta(-\lambda^\gamma)] \approx 1-\beta\sum_\gamma \Theta(-\lambda^\gamma)$, the integrations over λ imply that all terms which contain mixed replicas disappear. Thus Eq. (3.18) reduces to

$$1-\beta\sum_\gamma \left[\int Dr\, d\hat{\lambda}^\gamma \int_0^\infty \frac{d\lambda^\gamma}{2\pi} \exp\{ i\hat{\lambda}^\gamma[\lambda^\gamma + |r|(\mathbf{B}\cdot\mathbf{J}^\gamma)] - \tfrac{1}{2}(\hat{\lambda}^\gamma)^2[1-(\mathbf{B}\cdot\mathbf{J}^\gamma)^2] \right] = 1-\beta\sum_\gamma \int Dr\, H\left[\frac{|r|(\mathbf{B}\cdot\mathbf{J}^\gamma)}{\sqrt{1-(\mathbf{B}\cdot\mathbf{J}^\gamma)^2}} \right]$$
$$= 1-\sum_\gamma \frac{\beta}{\pi}\cos^{-1}(\mathbf{B}\cdot\mathbf{J}^\gamma) , \tag{3.21}$$

by a well-known identity. Here $H(x) \equiv \int_x^\infty Du$. However, the limit as p tends to infinity of $[1+(A/p)]^p$ is $\exp(A)$ for any A; so taking the $p \to \infty$ limit, with $\beta = N\tilde{\alpha}/p$, implies that

$$\langle Z^n \rangle_\xi = \left[\int d\mu(\mathbf{J})\, e^{(-N\tilde{\alpha}/\pi)\cos^{-1}(\mathbf{B}\cdot\mathbf{J})} \right]^n , \tag{3.22}$$

which is the partition function of n independent systems, each with an energy which is p times the generalization function of a binary perceptron, Eq. (2.16). Note that in moving between Eqs. (3.21) and (3.22) we have implicitly taken the limit $T \to \infty$ before taking $N \to \infty$. A more careful calculation gives the same result with the other, more natural, order of limits.

Following Eq. (2.14), we should also average $\langle Z^n \rangle_\xi$ over $P(\mathbf{B})$; but actually this is unnecessary. All realizations of $\mathbf{B}$ give the same result. This is because we have analyzed the special case in which the distribution of question vectors is uniform around the origin, and similarly $d\mu(\mathbf{J})$ is uniform on the unit sphere. That is, *all* directions are equally easy to learn.

Notice that this would not be the case if questions were correlated, or if $d\mu(\mathbf{J})$ were not uniform. In these cases the ease of learning does depend upon $\mathbf{B}$, and $P(\mathbf{B})$ must be averaged over. Further analysis of this expression is very elegant, but is deferred to Sec. III.C.1.

6. The annealed approximation

One useful approximate method is the so-called *annealed approximation:* $\langle \ln Z \rangle_\xi$ is approximated by $\ln\langle Z \rangle_\xi$. From Eq. (3.2) we can see that the annealed approximation is equivalent to representing the average of a quantity $A(\mathcal{N},V)$, that is, $\langle\langle\langle A(\mathcal{N},V)\rangle_{\mathcal{N}}\rangle_V\rangle_\xi$, by

$$\left\langle \left\langle \frac{\int d\mu(\mathcal{N})\, A(\mathcal{N},V) e^{-\beta E(\mathcal{N})}}{Z} \right\rangle_V \right\rangle_\xi \approx \frac{\int d\mu(\mathcal{N}) \langle\langle A(\mathcal{N},V) e^{-\beta E(\mathcal{N})}\rangle_V\rangle_\xi}{\langle\langle Z\rangle_V\rangle_\xi} .$$

Falk (1975) has pointed out that the annealed free energy is a lower bound for the correct free energy, but the minima of the annealed and correct (quenched) free energies are not always at the same place, i.e., both approaches do not always lead to the same equilibrium results. However, we shall give some examples below for which this approximation gives qualitatively correct answers.

The annealed approximation is equivalent to treating questions in the training set as annealed (rather than quenched) variables, such as **J**. This means, as Seung, Sompolinsky, and Tishby (1992) have pointed out, that a Langevin dynamics, Eq. (3.1), takes place in both the space of interactions and of examples.

At high temperatures the annealed approximation becomes exact and equivalent to the high-T theory. However, the work of Seung, Sompolinsky, and Tishby (1992) suggests that the annealed approximation is invalid for low temperatures and makes qualitatively incorrect predictions for the case of some unlearnable rules (Sec. V.A.4).

7. The "quenched" theory and the zero-temperature limit

Having outlined these two simplifying approaches, we now return to the full analysis of the problem at an arbitrary temperature.

The analysis can be continued by introducing the order-parameter matrix $q^{\gamma\gamma'} \equiv \langle \langle J^\gamma \cdot \mathbf{J}^{\gamma'} \rangle_\beta \rangle_\xi$ for $\gamma \neq \gamma'$, where the superscripts γ and γ' refer to different replicas of the system and are taken from the set $\{1, \ldots, n\}$. Here $\langle \cdots \rangle_\beta$ just means the thermal average over the Gibbs distribution for the replicated system. Similarly, $R^\gamma \equiv \langle \langle \mathbf{J}^\gamma \cdot \mathbf{B} \rangle_\beta \rangle_\xi$ measures the overlap with the teacher of the γth replica from the Gibbs distribution.

Since $\mathcal{H}$ is invariant under permutations of the replica indices, we usually assume the same applies to $q^{\gamma\gamma'}$, so that the *replica-symmetric* (RS) approximation, that $q^{\gamma\gamma'} = q$ for $\gamma \neq \gamma'$, would be exact.

Example: Continuing with the example of the binary perceptron, $\langle Z^n \rangle$ may be rewritten, using Eq. (3.18), as

$$\int \left[\prod_\gamma d\mu(\mathbf{J}^\gamma) dR^\gamma \right] \left[\prod_{\gamma<\gamma'} dq^{\gamma\gamma'} \right] e^{N\alpha h(\{q^{\gamma\gamma'}\},\{R^\gamma\})} \prod_{\gamma<\gamma'} \delta(q^{\gamma\gamma'} - \mathbf{J}^\gamma \cdot \mathbf{J}^{\gamma'}) \prod_\gamma \delta(R^\gamma - \mathbf{B} \cdot \mathbf{J}^\gamma) , \tag{3.23}$$

where h is defined from

$$e^h = \int \left[\prod_\gamma \frac{d\hat{\lambda}^\gamma d\lambda^\gamma}{2\pi} \right] Dr \exp\left[-\beta \sum_\gamma \Theta(-\lambda^\gamma) + i \sum_\gamma \hat{\lambda}^\gamma \lambda^\gamma - i \sum_\gamma \hat{\lambda}^\gamma |r| R^\gamma - \tfrac{1}{2} \sum_{\gamma\gamma'} \hat{\lambda}^\gamma \hat{\lambda}^{\gamma'} (q^{\gamma\gamma'} - R^\gamma R^{\gamma'}) \right] . \tag{3.24}$$

If we assume replica symmetry, $q^{\gamma\gamma'} = q$ for $\gamma \neq \gamma'$ and $R^\gamma = R$ for all γ, it is possible to evaluate h explicitly as a function of q, R, and n. Similarly, we may write the δ functions as the integrals

$$\int \frac{d\hat{R}}{2\pi} \frac{d\hat{q}}{2\pi} \exp\left[iN \sum_\gamma \hat{R}(R - \mathbf{B} \cdot \mathbf{J}^\gamma) + iN \sum_{\gamma<\gamma'} \hat{q}(q - \mathbf{J}^\gamma \cdot \mathbf{J}^{\gamma'}) \right] . \tag{3.25}$$

The $\mathbf{J}^\gamma$ are now integrated out. At this stage B disappears from the equations, and we do not need to average over it. This is for the reason given in Sec. III.A.5: for uniform $d\mu(\mathbf{J})$ and a uniform distribution of questions, all **B**-vectors are equally easy to learn. The $n \to 0$ limit is now taken and the free energy F is evaluated using the saddle-point equations for q, $\hat{q}$, R, and $\hat{R}$ as $N \to \infty$ (Györgyi and Tishby, 1990). The generalization error may then be found from R as function of α using Eq. (3.8). Thus learning has been completely and *exactly* solved; the results are given in Sec. III.C.1.

What is the physical interpretation of q? Noting that the replicas are independent systems evolving stochastically on the same energy landscape, it is usually argued (see Binder and Young, 1986) that $q = \langle \langle \mathbf{J} \rangle_\beta \cdot \langle \mathbf{J} \rangle_\beta \rangle_\xi$, which is a measure of how well constrained **J**'s chosen from the Gibbs distribution, Eq. (3.2), are by a typical set of examples.

At low temperatures, as for spin glasses, the coupling of different replicas, as in Eq. (3.18), leads to mathematical subtleties (Mézard *et al.*, 1987, is an excellent review). The symmetry group of $\mathcal{H}$—here the permutation group of replicas—may spontaneously break into a subgroup, which is called replica symmetry breaking (RSB). Its interpretation is that the space in which **J** evolves under the stochastic dynamics becomes disconnected and ergodicity is broken. Since the replicas are independent systems subject to the quenched noise, different replicas may become trapped in different regions, *pure states,* of the **J**-space. The average overlap of two replicas in the same pure state is different from that of two replicas in different pure states. In the one-step replica-symmetry-breaking ansatz the $n \times n$ order-parameter matrix $q^{\gamma\gamma'}$ acquires a special block structure of $m \times m$ submatrices, shown shaded in Fig. 7. The shaded blocks have diagonal elements 1 and off-diagonal elements of the order parameter q_1, which measures the overlap of pure states with themselves. The order parameter q_0, which forms all the elements of the off-diagonal blocks in Fig. 7,measures the overlap of two different pure states. The variables q_0, q_1, and m, as well as R and $\hat{R}$, must be determined self-consistently from the free-energy saddle-point equations.

As in spin glasses (de Almeida and Thouless, 1978), replica symmetry breaking is often indicated by one of the eigenvalues of the Hessian matrix (the matrix of second-order changes in the free energy for fluctuations of the order parameters) becoming negative, which implies that replica symmetry is unstable.

We shall only present results of RSB in one case, Sec.

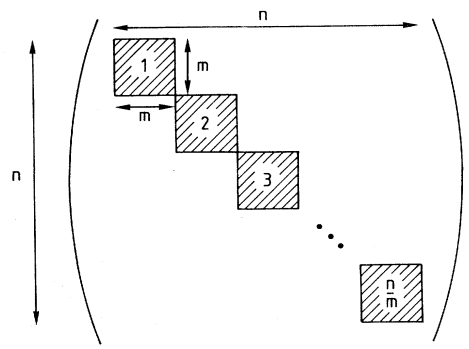

FIG. 7. Structure of the $q^{\gamma\gamma'}$ matrix with one-step replica symmetry breaking. The (n/m) submatrices are $m \times m$ blocks. Their diagonal elements are 1 and their off-diagonal elements are all q_1. Outside these blocks all elements are q_0.

III.C.1, since it can usually be shown that RS is exact (as in the case of the spherical binary perceptron in the example above) or at least a good approximation. Exactly correcting the results for RSB is believed to be always possible, but the calculation is usually very involved.

Using Eqs. (2.14), (2.13), (3.14), Seung, Sompolinsky, and Tishby (1992) gave an explicit form for ϵ_g, namely,

$$\epsilon_g = \lim_{n\to 0} \langle Z^{n-1} \int d\mu(\mathbf{J}) \epsilon_f(\mathbf{J}) e^{-\beta E(\mathbf{J})} \rangle_\xi$$
$$= \lim_{n\to 0} \int \left[\prod_\gamma d\mu(\mathbf{J}^\gamma) \right] \epsilon_f(\mathbf{J}^1) e^{-p\mathcal{H}(\{\mathbf{J}^\gamma\})} . \qquad (3.26)$$

From the definitions of ϵ_t and ϵ_g, one can show (Seung, Sompolinsky, and Tishby, (1992) that for all α and T, $\epsilon_t \leq \epsilon_g$, and that for large α, both approach the minimal value of the generalization error.

B. Sophisticated learning theory for a binary perceptron

In the preceding section we studied, as an example, learning with a binary perceptron by stochastically minimizing E_t at high and low temperatures. Before we present results, however, we shall describe two learning techniques which exploit the knowledge that the rule is linearly separable. Since in this case the rule space and student space are identical (both are the unit N-sphere), we may speak carelessly of a $\mathbf{J}$ which has successfully learned all the examples as lying in the version space, by which we mean that a $\mathbf{B}$ vector which equals $\mathbf{J}$ is in the version space.

1. The maximum stability algorithm

The contribution to the training error, $E_t(\mathbf{J})$, from example μ is 1 if and only if $\mathbf{J}$ gets the example wrong. However, it is possible to make this contribution more sophisticated by relating it to a physical quantity which measures "the certainty with which the student gets the question right," by which we mean the overlap $\Lambda^\mu \equiv \sqrt{N}\, \xi_o^\mu \mathbf{J}\cdot\boldsymbol{\xi}^\mu$. Making $E_t(\mathbf{J})=0$ simply means making all the set $\{\Lambda^\mu\}$ positive; so the $\mathbf{J}$ which it generates may lie anywhere in the version space. Referring to Fig. 8(a),

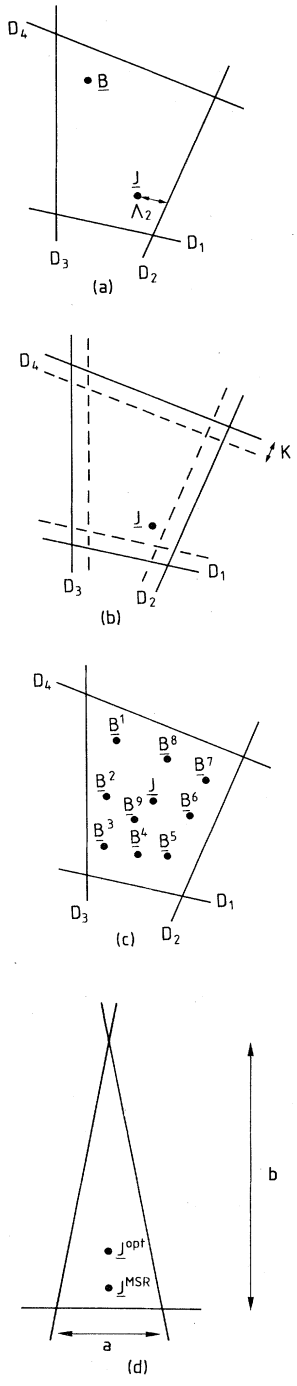

FIG. 8. Placing $\mathbf{J}$ within the version space: (a) A schematic diagram in the manner of Fig. 4(a). The version space is bounded by planes $\mathcal{D}_1$–$\mathcal{D}_4$. The perceptron vector $\mathbf{J}$ lies in the version space but far from the true rule $\mathbf{B}$. Λ_2 is the distance between $\mathbf{J}$ and plane $\mathcal{D}_2$. (b) Planes $\mathcal{D}_1$–$\mathcal{D}_4$ again a layer of thickness κ. (c) Samplers $\mathbf{B}^1$ to $\mathbf{B}^9$ are generated randomly in version space, and $\mathbf{J}$ is set to be their mean. (d) A two-dimensional section of the question space in which the version space happens to be narrow isosceles triangle, height b and width a. $\mathbf{J}^{\mathrm{MSA}}$ is $\sim a/2$ from the base and $\mathbf{J}^{\mathrm{opt}}$ is $\sim b/3$ away, near to far more of the triangle's area.

we see this is a little unfortunate, because a **J** near one side of the version space will have a low overlap with a **B** which happens to lie on the other side. However, Λ^μ in this diagram is the distance between **J** and the plane $\mathcal{D}_\mu$ perpendicular to ξ^μ; so to force **J** towards the center of the space we insist that the value Λ^μ for every μ is as high as possible, or, to be more exact, that the **J** which is generated has even the *lowest* value of Λ^μ greater than a value κ. Graphically, this means that **J** is at least κ away from the plane to which it is closest. Thus, as shown in Fig. 8(b), all the planes gain a "layer" of thickness κ, pointing in the direction of the version space, pushing **J** into a reduced volume in the middle. Clearly, as the value of κ rises, the size of the space available to **J** decreases until at some value of κ it shrinks to a point: the **J** vector with maximum stability $\kappa_{\max}$. If $\kappa_{\max}>0$, then **J** has correctly learned all given examples. This is the maximum stability algorithm (MSA); it is clearly an *ad hoc* approach to placing **J** within the version space, but as we shall see it is better than placing **J** there randomly (Opper *et al.*, 1990). It was inspired by a similar technique developed for the different problem, mentioned earlier, of storing memories in networks (Gardner, 1987, 1988).

The MSA could be implemented in the formalism above by minimizing an energy,

$$E_{\mathrm{MSA}}(\mathbf{J},\kappa)=\sum_\mu \Theta(\kappa-\sqrt{N}\,\xi_o^\mu \mathbf{J}\cdot\xi^\mu)\ ; \tag{3.27}$$

but it is equivalent in the zero-temperature limit to return from the canonical ensemble treatment (Gardner and Derrida, 1988) to the microcanonical formulation initiated by Gardner (1988) and analyze explicitly the fractional volume of the student space embedding every input-output configuration with a stability larger than κ, that is,

$$V_{\mathrm{frac}}=\frac{\int d\mu(\mathbf{J})\prod_\mu \Theta(\Lambda^\mu-\kappa)}{\int d\mu(\mathbf{J})}\ , \tag{3.28}$$

which shrinks to zero as κ rises to $\kappa_{\max}$. Since this is the fractional volume in an N-dimensional space, we expect that, as $N\to\infty$, the self-averaging quantity will be the extensive one, $\ln(V_{\mathrm{frac}})$. As in the macrocanonical approach, we require the method of replicas to perform the average of a logarithm.

Meir and Fontanari (1992) have recently extended this analysis to minimizing energies of the form

$$E_r=\sum_\mu (\kappa-\Lambda^\mu)^r\Theta(\kappa-\Lambda^\mu),\quad r=0,1,2\ . \tag{3.29}$$

For $r=0$, this reduces to E_{MSA}, Eq.(3.27). They find that, always selecting the best value for κ, $\epsilon_g(\alpha)$ decreases faster as r is increased beyond 0.

2. Optimal learning

Optimal learning, explained in Sec. II.G, is a more efficient way to exploit the prior information that the rule is linearly separable. It finds the **J** which agrees with as many rules in version space as possible on as many examples as possible. It may be generated by minimizing energy E_{opt}, Eq. (2.37), by the techniques of Sec. III.A or, as pointed out by Watkin (1991, 1993), since the rule space and version space coincide, by temporarily introducing m *sampler* points in version space, defined by vectors $\mathbf{B}^r$, $r=1,\ldots,m$. Each sampler is constructed independently by minimizing $E_t(\mathbf{B}^r)$, so that all lie randomly in the version space. We can then exploit the identity

$$\epsilon_n(\mathbf{J})=\langle \epsilon_f(\mathbf{J},\mathbf{B})\rangle_V=\lim_{m\to\infty}\frac{1}{m}\sum_{r=1}^{m}\epsilon_f(\mathbf{J},\mathbf{B}^r)\ , \tag{3.30}$$

showing that optimal learning is to generate $\mathbf{J}^{\mathrm{opt}}$, the **J** which minimizes the right-hand side of Eq. (3.30). This result is useful both as a way of generating the optimal **J** and as a tool for analyzing its success.

Replica symmetry is stable in zero-stability learning, so two randomly chosen networks which minimize $E_t(\mathcal{N})$ have a self-averaging overlap. It follows that two samplers, which are also points chosen randomly in the version space, have the *same* self-averaging overlap. Notice that this argument is using the second, Bayesian justification of statistical mechanics (Sec. III.A.3) as a way of measuring the size of version space. Using the self-averaging overlap of different samplers, it is easy to show (Watkin, 1993) that Eq. (3.30) may be rewritten

$$\epsilon_n(\mathbf{J})=\epsilon_f\left[\mathbf{J},\left[\lim_{m\to\infty}\frac{1}{m}\sum_{r=1}^{m}\mathbf{B}^r\right]\right]\ . \tag{3.31}$$

ϵ_f is a decreasing function of the product of its arguments; so

$$\mathbf{J}^{\mathrm{opt}}=\frac{1}{\gamma}\sum_{r=1}^{m}\mathbf{B}^r \tag{3.32}$$

in the spherical case (where γ is introduced as a normalizing constant), or such that $J_i^{\mathrm{opt}}=\mathrm{sgn}(\sum_r B_i^r)$ in the Ising case. In the limit of large m, but still $\mathcal{O}(1)$ (i.e., not rising with N), the optimal **J** is generated by construction, lying approximately in the centroid of the version space, Fig. 8(c). The samplers may now be discarded: they were used simply to train the perceptron optimally. Analogous constructions are used again, however, in Sec. VI.C.1, where they form part of a more complex network being built (Opper and Haussler, 1991, 1991b).

The justification for the optimal rule is much the same as for the maximum stability rule; the **J** generated lies as close as possible to as much as possible of version space. However, we can see that optimal learning is superior from Fig. 8(d), which shows a certain cross section of version space. In this cross section the version space happens to be bounded by three planes which form a narrow isosceles triangle, whose shortest side has width a and whose perpendicular height is b. Maximum stability produces a **J** only about $a/2$ from the base (the furthest point from all three sides), while the optimal rule produces one about $b/3$ up the triangle, close to far more possible rules.

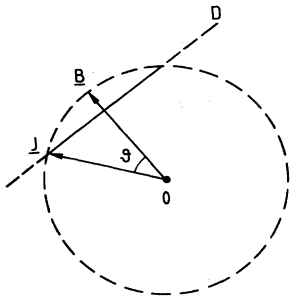

FIG. 9. The **B-J** plane. **J** and **B** lie on the unit N-sphere, centered at the origin $\mathcal{O}$. $\mathcal{D}$ is the hyperplane of all vectors whose overlap with **B** is R, and the projection into the plane of its intersection with the N-sphere is shown as in the solid part of line $\mathcal{D}$.

C. Results for perceptron learning

1. Learning a binary rule with a binary perceptron

a. The spherical perceptron

To illustrate the great mathematical ease and elegance of the high-temperature learning analysis, which was begun in the example of Sec. III.A.5, the rest will be explained in some detail. In Eq. (3.20) the variable on the right-hand side, which we have not yet calculated, is the entropy s. Figure 9 shows the projection of the N-dimensional space onto a two-dimensional hyperplane containing **J** and **B**, which lie on the unit N-sphere centered at the origin. The hyperplane $\mathcal{D}$, composed of all N-vectors whose overlap with **B** is R, intersects the unit N-sphere on an $(N-1)$-dimensional sphere, whose projection into the plane of the diagram is the solid part of the line $\mathcal{D}$. It is clear from the diagram that the radius of the small sphere is just $\sin\theta$. To within a constant the entropy per weight in the spherical case, $s_{\rm sp}$, is the logarithm of the $(N-2)$-dimensional surface area of this sphere and is given by

$$s_{\rm sp}(R)=\frac{N-2}{N}\ln(\sin\theta)\rightarrow\tfrac{1}{2}\ln(1-R^2)\ , \tag{3.33}$$

as $N\rightarrow\infty$. Thus the free energy to be minimized reads

$$\beta f=\frac{\tilde{\alpha}}{\pi}\cos^{-1}(R)-\tfrac{1}{2}\ln(1-R^2)\ , \tag{3.34}$$

a function possessing exactly one minimum which moves towards $R=1$ as $\tilde{\alpha}$ increases. Thus learning has been exactly solved. The generalization error decreases as $1/\tilde{\alpha}$ for large $\tilde{\alpha}$.

Zero-stability learning, the example developed in Sec. III.A.7 which placed **J** randomly in the version space, has been analyzed at zero temperature by Györgyi and Tishby (1990) and Sompolinsky *et al.* (1990). As we have said, it is equivalent to placing **J** randomly in the version space. Their solution is in terms of two physically meaningful order parameters q and R, which, as explained, may be calculated from Eq. (3.24): $R(\alpha)$, which measures the average overlap between **J** and **B**, and $q(\alpha)$, the overlap between two replicas, which is interpreted as the average overlap between two independent random choices of **J** in the version space; however, since **B** and all the replicas, $\{\mathbf{J}^\gamma\}$, are equally likely to lie anywhere in the version space, by symmetry $R(\alpha)=q(\alpha)$ (a result which can be verified from the thermodynamics). The resulting generalization error may be calculated from $R(\alpha)$ using Eq. (2.16) and is shown as line 1 in Fig. 10.

The maximum stability algorithm was analyzed by

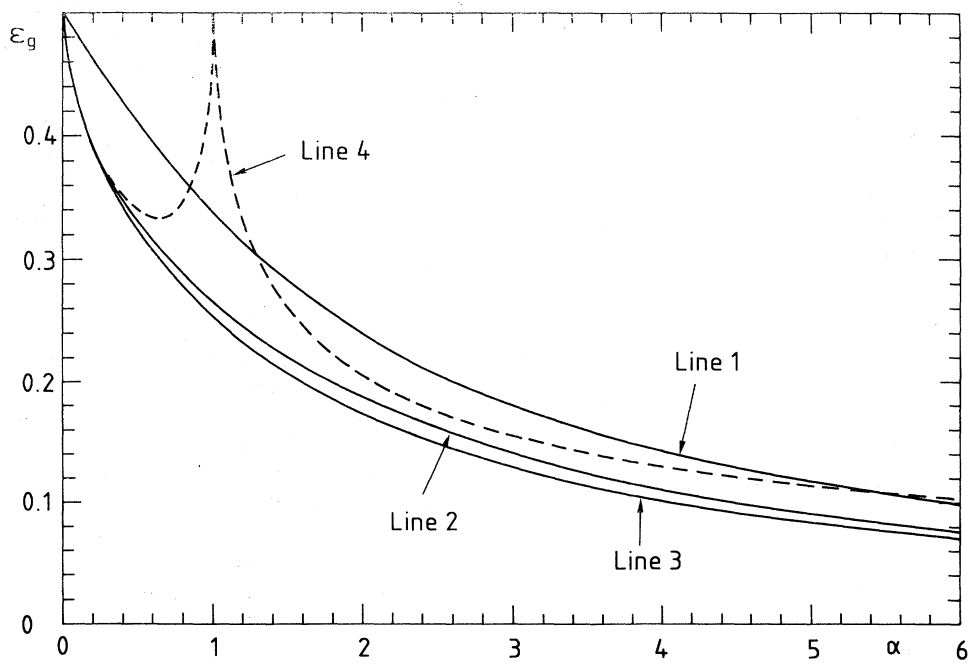

FIG. 10. Generalization error against α as a result of learning a linearly separable rule with a spherical perceptron with zero stability (line 1), using the MSA (line 2) and optimally (line 3). The pseudoinverse rule is shown dashed as line 4.

Opper *et al.* (1990), and optimal learning by Watkin (1991, 1993). As explained in Sec. III.B.2, the m samplers used in Eq. (3.32) to generate optimal learning are trained independently using the zero-stability technique, so that they have an overlap with $\mathbf{B}$ and with one another of the same $q(\alpha)$ found by the zero-stability calculation above. Calculating $\mathbf{J}^{\rm opt}\cdot\mathbf{J}^{\rm opt}$ using Eq. (3.32) therefore gives us the normalizing constant $\gamma=[m+m(m-1)q]^{1/2}$; thus, from Eq. (3.31),

$$\epsilon_g=(1/\pi)\cos^{-1}\left[(1/m\gamma)\left[\sum_{r=1}^{m}\mathbf{B}^r\right]\cdot\left[\sum_{r=1}^{m}\mathbf{B}^r\right]\right]$$
$$\rightarrow(1/\pi)\cos^{-1}[\sqrt{q(\alpha)}] \qquad (3.35)$$

as m becomes large [but still $\mathcal{O}(1)$]. Remarkably, the optimal perceptron generalizes for the linearly separable rule as well as the Bayes algorithm, described in Sec. II.F, because the proportion of examples on which they differ is vanishingly small. This means that *in the large N limit a spherical perceptron can learn a linearly separable rule as well as any other possible network.*

The results of these algorithms are shown in Fig. 10 as lines 2 and 3, respectively. In all cases the average generalization error decreases as $1/\alpha$, but with a lower constant for the maximum stability approach and a still lower one for the optimally taught perceptron. In all these models replica symmetry is stable against small fluctuations (Györgyi and Tishby, 1990) and the solutions are therefore thought to be exact for all α and T.

Simulations lend strong support for all these analytical conclusions. Instead of explicitly minimizing $E_{\rm MSA}$, it is faster to use an equivalent algorithm, for example, the MinOver algorithm (Krauth and Mézard, 1987) or the Adatron algorithm (Anlauf and Biehl, 1989), which iterates any random vector towards the maximally stable $\mathbf{J}$. Interestingly, if the generalization error is calculated at each stage of the process, it actually falls below the result predicted for the MSA by thermodynamics, before rising back eventually to the thermodynamic prediction as the number of iterations becomes large (G.-J. Bex, private communication, 1992). This is an example of overfitting, discussed in Sec. VI.C.3.

Optimal learning was recently successfully simulated by Watkin (1993) using $m=25$; in this simulation a trick was employed to generate samplers more quickly than by independent stochastic minimization of $E_t(\mathbf{B}^r)$.

We might mention that other functions of $\{\Lambda^\mu\}$ have been used as error measures. The *pseudoinverse* rule (Personnaz *et al.*, 1986; Vallet *et al.*, 1989) tries to give all the Λ^μ the same constant value ρ. This is possible for $\alpha\le 1$, and in this range a unique $\mathbf{J}$ is generated by making ρ as high as possible. The process may be generalized for higher α by minimizing an energy which is the sum over the training set of the quadratic deviation of $\{\Lambda^\mu\}$ from ρ, $E_{\rm pseud}\equiv\sum_\mu(\Lambda^\mu-\rho)^2$; again there are a unique ρ and $\mathbf{J}$ which minimize this expression. The zero-temperature analysis of such a rule for the spherical perceptron has been performed (Opper *et al.*, 1990), and the results are shown in Fig. 9 by dashed line 4. At $\alpha\approx 1$, there is strong overfitting: the only $\mathbf{J}$ which has the same overlap with all examples has no correlation with $\mathbf{B}$; thus $\mathbf{J}$ depends entirely upon the random details of questions in the training set. As $\alpha\rightarrow\infty$, $\epsilon_g\sim 1/\sqrt{\alpha}$, like the Hebb algorithm, Sec. II.D. In fact, a cost function linear in the Λ^μ just leads to the Hebb algorithm, Eq. (2.17), for $T=0$ (Wong and Sherrington, 1990).

b. The Ising perceptron

The high-temperature analysis for the Ising perceptron is equally straightforward (Sompolinsky *et al.*, 1990) for learnable problems (those in which every component of $\mathbf{B}$ is ± 1). The corresponding entropy can be obtained using a simple counting argument: if $\mathbf{B}\cdot\mathbf{J}=R$, then any given component i of the weight vector must have chance $(1+R)/2$ of $B_i=J_i$ and $(1-R)/2$ of $B_i=-J_i$. Thus

$$s_{\rm Is}(R)\equiv-\sum_\lambda p_\lambda\ln p_\lambda$$
$$=-\left[\frac{1+R}{2}\right]\ln\left[\frac{1+R}{2}\right]$$
$$-\left[\frac{1-R}{2}\right]\ln\left[\frac{1-R}{2}\right], \qquad (3.36)$$

where λ labels the possibilities for each node. As long as $\tilde{\alpha}<\tilde{\alpha}_{\rm th}\approx 1.69$, the resultant free energy has two minima, one at $R=1$ and a lower one at $0<R<1$. For $\alpha>\tilde{\alpha}_{\rm th}$, the minimum at $R=1$ is the global one, but the minima are still separated by an energy barrier scaling with N. Thus, starting without initial information ($R\approx 0$), the student would evolve to the minimum at $0<R<1$ and then take exponentially long to reach the minimum at $R=1$. However, for $\tilde{\alpha}$ greater than a *spinodal* value of $\tilde{\alpha}_{\rm sp}\approx 2.08$, the minimum at $0<R<1$ disappears and the system converges fast to the state at $R=1$. It is remarkable that this first-order transition to perfect generalization exists even at high temperatures, but this was confirmed in simulations by Sompolinsky *et al.* (1990), as shown in Fig. 11(a), where the solid line shows the high-temperature prediction and the small boxes show the results, and error bars, of a simulation using $N=75$ and $T=5$. The vertical dashed line is at $\tilde{\alpha}_{\rm th}$, where the network with perfect generalization becomes the global free-energy minimum. The results from the annealed approximation are qualitatively similar to those of the high-T approach.

It is worth pointing out here that finite-size effects mean that at high temperatures a barrier between two free-energy minima may be climbed if the stochastic dynamics is given a long enough time in which to evolve (which would not be possible as $N\rightarrow\infty$, which makes the barrier infinitely high). This effect has been observed in simulations by Kocher and Monasson (1991) and by Schwarze, Opper, and Kingel (1992) and will be referred to again in Sec. VI.C.1.

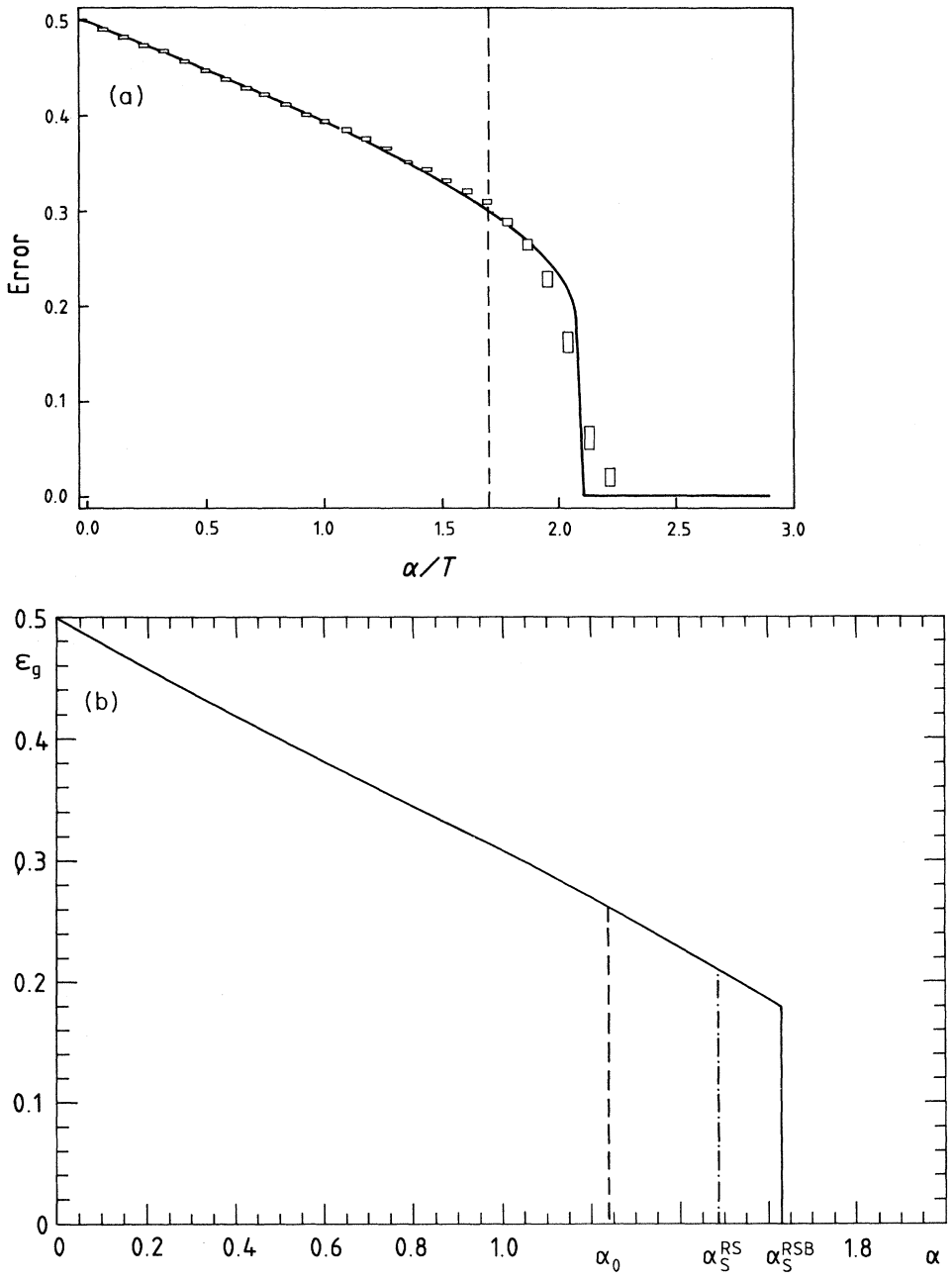

FIG. 11. Learning with an Ising perceptron: (a) The solid line shows the high-temperature prediction of the generalization error against α. The dashed vertical line shows the position of the thermodynamic transition. The boxes show the results and error bars of a simulation (Sompolinsky *et al.*, 1990) with $N=75$ at $T=5$. (b) The generalization error for zero-temperature learning, showing the three predictions for the transition to perfect generalization.

At zero temperature and zero stability the annealed approximation and the replica-symmetric approach were analyzed by Györgyi (1990b). A more comprehensive analysis was performed by Seung, Sompolinsky, and Tishby (1992). We first report on the replica-symmetric results. For $\alpha<\alpha_{\rm th}\approx 1.245$ the RS free energy has two minima—a result first derived by Gardner and Derrida (1989)—one at $R=1$ and one at $0<R<1$. The solution at $0<R<1$ has the lower free energy and is therefore the equilibrium solution. For $\alpha<\alpha_{\rm th}$ there are many states with zero training error, and most of them have an overlap with the teacher of $0<R<1$. As α is increased beyond $\alpha_{\rm th}$, the solution at $R=1$ becomes the absolute minimum; however, the solution at $0<R<1$ persists until the spinodal point of $\alpha_{\rm sp}^{\rm RS}\approx 1.49$ is reached. For $\alpha_{\rm th}<\alpha<\alpha_{\rm sp}^{\rm RS}$ stochastic algorithms which iterate from a random initial configuration ($R\approx 0$) will not converge to the global minimum of free energy, but become stuck in one of the metastable states. This part of the phase diagram is interpreted as a spin-glass phase (Mézard *et al.*, 1987) rich in metastable states.

Since a system with a finite number of discrete states cannot have a negative entropy, the line in the α-T space, indicating where the entropy becomes negative, provides

a lower bound for temperatures below which the RS assumption is incorrect. Indeed, it was negative entropy which gave the first clue that replica symmetry was sometimes untrue for spin glasses (Sherrington and Kirkpatrick, 1975). If answers are uncorrelated with questions (the storage of memories), the zero-entropy line is exactly the line on which replica symmetry breaking occurs (Krauth and Mézard, 1989). Following their approach, Seung, Sompolinsky, and Tishby (1992) have analyzed the first-step replica symmetry breaking in accordance with the Parisi theory (reviewed by Mézard *et al.*, 1987, and described in Sec. III.A.7). For small enough temperatures there is a metastable spin-glass phase. The thermodynamic transition at zero temperature is still at $\alpha_{\rm th}\approx 1.245$, whereas the spinodal line indicating the disappearance of the spin-glass phase is shifted to $\alpha_{\rm sp}^{\rm RSB}\approx 1.63$. The predictions of generalization error are shown on Fig. 11(b), which indicates the three different predictions of the value of α at which the transition to perfect generalization occurs.

The interpretation of these results is that the student space for the Ising perceptron becomes disconnected with a central region, in which ergodicity is unbroken and which contains a network which generalizes perfectly ($R=1$), surrounded by regions in which ergodicity is broken. The sheer bulk of the outer regions, rising as $\exp[Ns_{\rm Is}(R)]$, means that in these regions will be networks such that small changes to the network within the student space always increase the number of questions in the training set which are answered incorrectly. These networks are the metastable states associated with RSB. Passing between the fully ignorant network, $R\approx 0$, and the highly taught network $R\approx 1$, is liable to become exponentially slow because the dynamics leads **J** through one of the outer regions. Metastable states are far more rare for spherical perceptrons than for Ising ones, because for the former the student space is continuous and so allows far more freedom of movement; therefore replica symmetry is unbroken.

Recently Derrida, Griffiths, and Prügel-Bennett (1991) investigated the finite-size effects of learning with binary Ising perceptrons. By numerically enumerating all the possibilities for **J**, they observed that, for finite N ($N\approx 20$), learning from examples in which each component of the questions is chosen from a Gaussian distribution differs strongly from the case in which components of the questions are randomly ± 1, although, as pointed out in Sec. II.D, for $N\to\infty$ the two cases give the same result (since in both cases the central limit theorem applies to the overlap between **B** and questions). They find, in addition, that the critical number of patterns beyond which a sudden transition to perfect generalization occurs depends strongly and in a nonlinear way upon N, which makes it very difficult to extrapolate numerical results to the infinite N limit. Using a solvable "toy" model, however, they derive an exact upper bound for the thermodynamic transition, namely, $\alpha_{\rm th}^{\rm up}\approx 1.45$, which is obviously in agreement with the result of the replica method, $\alpha_{\rm th}=1.245$.

2. Learning a linear rule with a linear perceptron

Here we apply learning from examples to a linear perceptron whose output is given by $\mathcal{N}(S)=\sqrt{N}\,\mathbf{B}\cdot\mathbf{S}$. The questions $\{\xi^\mu\}$ and answers $\{\xi_o^\mu\}$ of the training set are related via the rule $\xi_o^\mu=\sqrt{N}\,\mathbf{B}\cdot\xi^\mu$. In Eq. (2.11) each pattern which is not correctly answered makes a contribution to the training error of 1, but in this case it is more natural to use the quadratic deviation between student and teacher output as error measure $e(\mathbf{J},\mathbf{B},\mathbf{S})$. The generalization function thus becomes (Hertz *et al.*, 1991)

$$\epsilon_f(\mathbf{J},\mathbf{B})=\left\langle (N/2)(\mathbf{J}\cdot\mathbf{S}-\mathbf{B}\cdot\mathbf{S})^2\right\rangle_{\rm S}, \tag{3.37}$$

which equals $(1-R)$, where $R\equiv\mathbf{J}\cdot\mathbf{B}$, if **J** is normalized by Eq. (2.8) and, again, components of questions are assumed to be drawn independently from a distribution independent of **B** and **J**.

a. The spherical perceptron

The problem of learning form examples in a linear spherically constrained perceptron was first solved by Krogh and Hertz (1991), who derived the whole dynamics of learning (see Sec. IV). The equilibrium solution was comprehensively analyzed by Seung, Sompolinsky, and Tishby (1992). At zero temperature the solution of the problem is straightforward, since it reduces to the solution of a set of linear homogeneous equations. Replica-symmetric theory is exact for all T and α and yields a linear decrease of ϵ_g from 1, for $\alpha=0$, to $\epsilon_g=0$ for $\alpha=1$, reflecting the fact that N linearly independent equations of the form $\sqrt{N}\,\mathbf{B}\cdot\xi=\xi_o$ determine **B**. At any finite temperature, however, this transition does not exist and the asymptotic result for smooth networks is $\epsilon_g\sim T/2\alpha$ (cf. Sec. VI.D).

The annealed approximation leads to the correct asymptotic behavior of ϵ_g for $T>0$. For $T=0$, however, it predicts $\epsilon_g\sim(2-2\alpha)/(2-\alpha)$, which is obviously not in agreement with the RS prediction reported above.

b. The Ising perceptron

In the case of the linear Ising perceptron (Seung, Sompolinsky, and Tishby, 1992), the values of B_i and J_i are constrained to be ± 1. In the high-T limit the average generalization error decreases exponentially quickly, i.e., $\epsilon_g\sim 2e^{-2\alpha}$. At zero temperature, using the same notation as for the binary Ising perceptron, we find $\alpha_{\rm th}=0$ and $\alpha_{\rm sp}^{\rm RS}\approx 0.48$. The result $\alpha_{\rm th}=0$ suggests that for any macroscopic number of examples the free energy has a global minimum at $R=1$, and the spinodal point $\alpha_{\rm sp}^{\rm RS}$ indicates the disappearance of metastable solutions at $0<R<1$. Monte Carlo simulations seem to support the results for temperatures as low as $T=0.2$, although finite-size effects result in slight deviations from the analytical theory.

The results reported here were obtained within the

framework of replica symmetry, but the position of the zero-entropy line shows that they are incorrect for low temperatures; it is expected that full RSB will shift the position of the spinodal line.

For the linear perceptron it is possible to calculate the density of local minima, which are separated by barriers of finite height but may still strongly influence the zero-temperature dynamics. Developing the work of Gardner (1986), Seung, Sompolinsky, and Tishby (1992) have calculated an upper bound for the number of local minima as a function of R. They find that there are no local minima in the neighborhood of the $R=1$ solution, implying that the energy function must be smooth there. With increasing α the dynamical attractor radius increases until at $\alpha \approx 2.39$ all local minima disappear, suggesting that even at $T=0$ the learning dynamics will be fast.

IV. DYNAMICS OF LEARNING

To make use of neural networks we need to know not just that there is a network which learns the rule, but also that we can find it on a realistic time scale. After all, many optimization problems are NP, which stands for nondeterministic polynomial. This means that no algorithm can find the solution in a time scaling less than exponentially with the size of the system—which is clearly not much use. A mathematically more rigorous introduction to combinatorial optimization problems is given by Garey and Johnson (1979) and Mézard *et al.* (1987, pp. 307–335). We first discuss an example of learning which is definitely *not* NP and for which the speed of algorithms may be calculated exactly. Then briefly we discuss learning in situations which *are* conjectured to be NP.

Even more importantly for physicists, this theory is applicable to disordered physical systems, since, as observed in Sec. III.A, gradient descent maps to zero temperature Langevin dynamics on a noisy energy landscape in the limit of high viscosity (Parisi, 1988), or on a noisy *velocity-dependent* energy surface. There are also close relationships to the dynamics of spin glasses, still an area of research (for example, Hammann *et al.*, 1992). While NP problems correspond to exponentially slow relaxation, the other case is exactly solvable noisy dynamics.

A. Exactly solved dynamics: the linear perceptron

Krogh and Hertz (1991) studied the dynamics of learning of a linear perceptron, introduced in Sec. II.A.2, whose error measure e was the quadratic deviation of the student answer to inputs and the correct answer, so that, for example,

$$E_t(\mathbf{J})=\tfrac{1}{2}\sum_{\mu}(\mathbf{J}\cdot\xi^{\mu}-\xi_o^{\mu})^2 . \tag{4.1}$$

Gradient descent using $E_t(\mathbf{J})$ means the noiseless version of Eq. (3.1), which is called Adaline learning (Widrow and Hoff, 1960). If the components of $\mathbf{J}$ are allowed to take any real value, a simple gradient descent can be guaranteed to find the global minimum of this convex function,

$$\frac{\partial \mathbf{J}}{\partial t}=-\nabla_J E_t(\mathbf{J})=A(\mathbf{B}-\mathbf{J}) , \tag{4.2}$$

where the elements of the $N\times N$ matrix A are given by $A_{jk}=\sum_{\mu}\xi_j^{\mu}\xi_k^{\mu}/N$. Defining the vector $\mathbf{D}\equiv\mathbf{B}-\mathbf{J}$, Eq. (3.37) shows that $\langle \epsilon_f \rangle_{\mathcal{N}(t)}=\frac{1}{2}\mathbf{D}(t)\cdot\mathbf{D}(t)$, where $\langle \cdots \rangle_{\mathcal{N}(t)}$ means the average over the network generated by this algorithm after a time t. Equation (4.2) gives $\partial\mathbf{D}/\partial t=-A\mathbf{D}$, the diffusion equation. Writing the normalized eigenvectors of A as $\{\mathbf{x}_l\}$, where $l=1,\ldots,N$, and its eigenvalues as $\{\lambda_l\}$, we can decompose $\mathbf{D}(t)=\sum_l y_l(t)\mathbf{x}_l$, where $y_l(t)\equiv\mathbf{D}(t)\cdot\mathbf{x}_l$, which implies that $\epsilon_f(t)=\sum_l y_l^2(t)/2$ and the differential equations become $\partial y_l(t)/\partial t=-\lambda_l y_l(t)$. Thus the components of ϵ_f relax independently and exponentially, with a time constant $2\lambda_l$. If the initial $\mathbf{J}$, $\mathbf{B}$, and A are uncorrelated, then each $y_l(0)$ will be Gaussian distributed with $\langle y_l(0)\rangle=0$ and $\langle y_1^2(0)\rangle=2/N$. As N becomes large we may therefore convert the sum over eigenvalues into an integral and write

$$\langle \epsilon_n(\mathcal{N})\rangle_{\mathcal{N}(t)}=\int d\lambda\,\rho(\lambda)\exp(-2\lambda t) , \tag{4.3}$$

where $\rho(\lambda)$ is the eigenvalue spectrum of A. To find $\epsilon_g(t)$ we only need average over the questions, so determining $\langle\rho(\lambda)\rangle_{\xi}$. Hertz, Krogh, and Thorbergsson (1989) calculated this quantity by a diagrammatic expansion, but we shall present the alternative approach of Opper (1989) and Kinzel and Opper (1991), which uses the technique of Edwards and Jones (1976) for calculating the spectrum of large, symmetric, random matrices. It uses the identity

$$\delta(\lambda-\mu)=\frac{1}{\pi}\mathrm{Im}\left\{\lim_{\epsilon\to 0}\left[\frac{1}{\lambda+i\epsilon-\mu}\right]\right\} . \tag{4.4}$$

Thus we can write

$$\begin{aligned}\rho(\lambda)&=\frac{1}{N\pi}\mathrm{Im}\left\{\lim_{\epsilon\to 0}\sum_l\left[\frac{1}{\lambda+i\epsilon-\lambda_l}\right]\right\}\\&=\frac{-2}{N\pi}\mathrm{Im}\left\{\lim_{\epsilon\to 0}\frac{\partial}{\partial\lambda}\ln Z(\lambda+i\epsilon)\right\} ,\end{aligned} \tag{4.5}$$

where

$$\begin{aligned}Z(\lambda)&=\left[\prod_l(\lambda-\lambda_l)\right]^{-1/2}\\&=(\det[\lambda I-A])^{-1/2}\\&=\left[\frac{i}{\pi}\right]^{N/2}\int dx\, e^{-i\mathbf{x}\cdot[\lambda I-A]\cdot\mathbf{x}} ,\end{aligned} \tag{4.6}$$

by a well-known identity, where I is the identity matrix. It is now possible to recognize Z as the partition function of a system at an *imaginary* temperature. Section III.A showed how to calculate $\langle \ln Z\rangle_{\xi}$; so it is straightforward

to obtain the average eigenvalue spectrum, which is sketched in Fig. 12: a δ function peak at zero of magnitude $(1-\alpha)\Theta(1-\alpha)$, which represents the asymptotic value to which ϵ_g relaxes as $t\to\infty$ (see Sec. III.C.2) and a continuum of values between η_1 and η_2, given by $\eta_{1,2}=(1\mp\sqrt{\alpha})^2$. The long-time behavior of $\epsilon_g(t)-\epsilon_g(\infty)$ is given by the smallest nonzero eigenvalue of A, that is, η_1. For any $\alpha\neq 1$, the error relaxes exponentially, whereas for $\alpha\to 1$, which implies $\eta_1\to 0$, the relaxation time diverges like $(\sqrt{\alpha}-1)^{-2}$, which is *critical slowing down* of the dynamics at the transition to perfect generalization. These results were rederived by Le Cun *et al.* (1991).

Krogh and Hertz (1991) considered various modifications of the training procedure, Eq. (4.2), by, for example, imposing onto the components of the weights a tendency to decrease (weight decay):

$$\frac{\partial \mathbf{J}}{\partial t}=A(\mathbf{B}-\mathbf{J})-\kappa\mathbf{J}\ . \tag{4.7}$$

Taking $\kappa>0$ removes the divergence of the relaxation time, but implies that relaxation of ϵ_g is never to zero. The asymptotic generalization error remains nonzero for any finite α.

Static noise on the weights or the output has also been incorporated in the model (Krogh and Hertz, 1991; Krogh, 1992). Adaline learning may also be solved in some regimes for a binary spherical perceptron (Opper, 1989; Kinzel and Opper, 1991), and this analysis has also been extended to more complex, nonlinear perceptron-learning schemes by Biehl, Anlauf, and Kinzel (1991).

Numerical algorithms do not run in continuous time, so it is necessary to invent a discrete version of the dynamics in Eq. (4.2):

$$\mathbf{J}(t+1)=\mathbf{J}(t)-\gamma\nabla_{\mathbf{J}}E_t(\mathbf{J}(t))\ , \tag{4.8}$$

where the parameter γ measures how large a step is made. Equation (4.8) implies that $\mathbf{D}(t+1)=[I-\gamma A]\mathbf{D}(t)$, so that the bracket is an operator representing one update of the network. Thus decomposing $\mathbf{D}$ once more into eigenvectors of A, we obtain after τ time steps that $y_l(\tau)=(1-\gamma\lambda_l)^\tau y(0)$, which falls to zero exponentially for τ large, providing that $|1-\gamma\lambda_l|<1$; thus we obtain the same exponential decay.

Since the eigenvalue spectrum derived above contains no values greater than $\mathcal{O}(1)$, for $|1-\gamma\lambda_l|^\tau$ to converge to zero γ may still be chosen to be $\mathcal{O}(1)$ (Opper, Diederich, and Anlauf, 1988; Opper, 1989; Kinzel and Opper, 1991). Unfortunately, if the questions we must learn are partially correlated (which will be true in practical applications), an additional eigenvalue $\mathcal{O}(N)$ appears in the spectrum (Edwards and Jones, 1976; Wendemuth, 1991). For the algorithm to converge, γ must be chosen to be $\mathcal{O}(1/N)$, which seriously slows down learning. Even a pattern correlation $\sum_j \xi_j^\mu \xi_j^{\mu'}$ of $\mathcal{O}(\sqrt{N})$ for $\mu\neq\mu'$, which is the magnitude of correlations we would expect to occur naturally if the questions *are* uncorrelated, generates this eigenvalue (Wendemuth, 1991); so in any practical problem one should take $\gamma\sim 1/N$, or use the sequential version of Adaline learning (Diederich and Opper, 1987), which has similar dynamics.

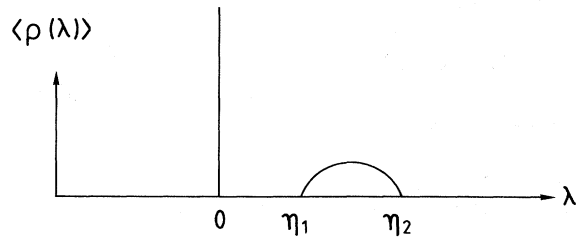

FIG. 12. The average eigenvalue spectrum of Sec. IV.A, $\langle\rho(\lambda)\rangle$, showing a δ peak at zero and a continuum of values between η_1 and η_2.

B. A dynamic mean-field theory approach: the binary Ising perceptron

Horner (1992a, 1992b) has presented an interesting dynamical approach to the problem of learning a rule and storing memories (random input-output configurations) in a binary Ising perceptron. His dynamic mean-field theory is based on the dynamical approach to disordered systems, which is a well-known alternative to the replica method (De Dominicis, 1978).

The process of learning a rule is analyzed directly as a random walk in student space according to a training energy E_t [Eq. (2.11)], in which the error measure has been generalized to

$$e(\mathcal{N},\xi_o^\mu,\xi^\mu)\equiv(-\Lambda^\mu)^r\Theta(-\Lambda^\mu)\ , \tag{4.9}$$

where, as previously, $\Lambda^\mu\equiv\sqrt{N}\,\xi_o^\mu\mathbf{J}\cdot\xi^\mu$. The parameter r can take all non-negative values. Setting $r=0$ just gives the energy used to train a binary Ising perceptron in Sec. III.C.1.b; however, because the landscape is locally flat for $r=0$, simulated annealing always leads to a freezing of the system and $r=0$ is therefore not used in this section.

Figure 13 shows the phase diagram for $r=1$. The

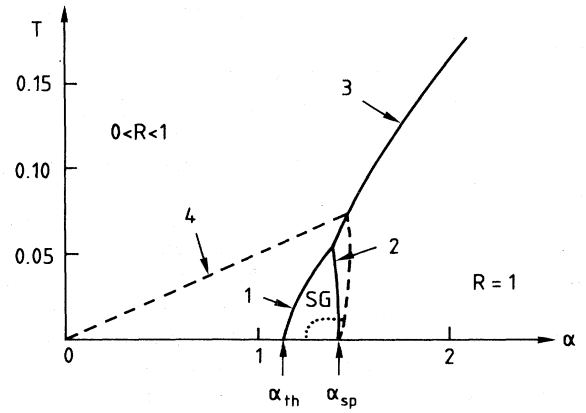

FIG. 13. Phase diagram for stochastic learning of an Ising linearly separable rule with an Ising perceptron using a smoothed energy ($r=1$). The solid lines show the results of an exact thermodynamic calculation, and the dashed lines show the results of an exact dynamic one.

solid lines show the results of the thermodynamic calculation, closely resembling those for $r=0$, which was discussed in Sec. III.C.1.b. To the left of line 1 thermodynamics predicts a value of R just greater than zero; replica symmetry is stable and the perceptron is unfrozen; for zero temperature the network is predicted to be in a state with $E_t(\mathbf{J})=0$. To the right of line 2 the perceptron has an $R\sim 1$ and gives $R\to 1$ at $T\to 0$. Crossing line 3 there is a discontinuous transition: R jumps from a small value to one near to 1. For all $\alpha>2.07$, R rises smoothly to 1 as $T\to 0$. There is also a spin-glass phase with first-step RSB between lines 1 and 2, and in this region thermodynamics suggests that first-step RSB is exact. Thus at zero temperature and rising α the perceptron moves from an unfrozen $0<R<1$ solution into a frozen RSB solution at line 1 (at a value of α which, as in the $r=0$ case, coincides with $\alpha_{\rm th}$), and then at the RSB spinodal point of $\alpha_{\rm sp}^{\rm RSB}=1.53$, freezes discontinuously into $R=1$. All this agrees fairly well with the thermodynamic results for $r=0$.

Unfortunately, it seems that these results are incorrect in several ways. The principal conclusion of Horner's exact dynamical treatment (1992b), strongly supported by numerical simulation (Horner, 1993; Patel, 1992), is that if a system at a given value of α is annealed to a low temperature, it experiences ergodicity breaking when it enters the larger region marked with the dashed line 4. This nonergodic phase is characterized by the existence of infinite relaxation time scales, when the system size becomes infinite. The resulting anomalous contributions to the response functions are related to correlation functions by some quasi-fluctuation-dissipation theorem. This scenario is analogous to first-step RSB.

At a lower temperature still, the system enters the region marked with the dotted line. There the entropy of the marginally stable solution vanishes, which can be considered as the temperature where first-step RSB becomes unstable and additional diverging time scales appear. These results indicate that although thermodynamics predicts that a Gibbs process at $T=0$ will produce states with zero energy for $\alpha<\alpha_0$ simulated annealing will fail for large systems for all $\alpha<\alpha_{\rm sp}$.

The explanation for this phenomenon seems to be (Horner, 1992b) that thermodynamics is demonstrating *existence proofs* for zero-energy solutions, while the dynamics theory tells us what is actually *observed* in a simulation. Note that the $r=1$ energy function is smoother than $r=0$, so one might expect RSB effects to be even greater for the $r=0$ case, where no dynamic solution is available. The moral seems to be that thermodynamics must be carefully applied; it may not be as successful as has been suggested at predicting ergodicity breaking. Horner (1992a) has conjectured that the maximum of the free energy determines the existence of perfect solutions, whereas the minimum yields information about the performance of a polynomial-time algorithm.

Note that both the dynamic and the thermodynamic solution predict that at least for $\alpha>1.245$ there *is* a unique $\mathbf{J}$ with zero energy—and this $\mathbf{J}$ must be $\mathbf{B}$. Dynamics and thermodynamics also agree that a stochastic process will not converge to this solution until $\alpha\sim 1.53$. Horner (1992b) has therefore conjectured that for α slightly greater than $\alpha_{\rm th}$, learning is a NP problem: *no* polynomial-time algorithm can converge to the unique right answer. However, as α rises to $\alpha_{\rm sp}^{\rm RSB}\approx 1.53$, the stochastic algorithm does converge in polynomial time; so the problem is certainly not NP. Is there therefore, we might speculate, a transition for $1.25<\alpha<1.53$ at which "NP-ness" disappears, arising because there is a single known solution into which enough examples will force $\mathbf{J}$? This question is particularly interesting, since all NP problems are in a sense equivalent (Garey and Johnson, 1979). The question of which sets of a data give NP difficulty is an open one in computational science, and this problem may suggest that a stochastic approach is appropriate.

Finally, Horner (1992b) has observed an interesting hysteresis effect. As observed, stochastic dynamics starting from high temperature experiences freezing on crossing the dashed line. However, if we begin at $T=0$ and $R=1$ (i.e., $\mathbf{J}=\mathbf{B}$) and *raise* the temperature, the perceptron only becomes *unfrozen* at a temperature higher than the one where it crosses the dashed line.

A similar effect has been observed in high-temperature learning by Kocher and Monasson (1991; and by Schwarze, Opper, and Kinzel (1992), whose results will be presented in context in Sec. VI.C.1). In high-T learning, the hysteresis effect is due to the competition of two local minima of the free energy at different values of the order parameter R. As discussed in Sec. III.C.1.b, starting from a random state ($R\approx 0$) and no examples in the training set, $\mathbf{J}$ evolves randomly. If new examples are added, and at every stage $\mathbf{J}$ is given time to evolve, then at a certain value of $\tilde{\alpha}=\tilde{\alpha}_{\rm th}$, as explained in Sec. III.C.1, the high-R solution becomes the global free-energy minimum, but the freezing transition, in which the low-R free-energy minimum disappears and there is a transition to the $R=1$ solution of perfect learning, occurs at a higher value of $\tilde{\alpha}=\tilde{\alpha}_{\rm sp}$. If $\tilde{\alpha}$ is decreasing, however, which corresponds to the somewhat unphysical case of a stochastic dynamics continuing while the training set is reduced, the reverse transition, from the $R=1$ solution to the low-R one, occurs when the high-R free-energy minimum disappears, which is at a lower value of $\tilde{\alpha}$ than $\tilde{\alpha}_{\rm th}$. Thus there is a *memory* effect.

This phenomenon is expected to occur in other situations with a first-order transition in learning (Sompolinsky and Tishby, 1990; Watkin and Rau, 1992b). Section VI.C points out that it is associated with interesting finite-size effects.

V. INCORPORATION OF PRACTICAL CONSIDERATIONS

While the results of Secs. III and IV are elegant and useful for gaining an insight into learning, we are

unaware of any practical situation in which their conclusions apply exactly. Here we describe four major ways in which practical problems are richer than these bare results and demonstrate how the theory may be suitably adapted.

A. Unlearnable rules

Since linearly separable rules form a very small class of all possible rules, it is of great interest, as Gutfreund and Toulouse (1992) have pointed out, to understand how perceptrons generalize in unlearnable problems. Nabutovsky and Domany (1991) have recently proposed an algorithm which converges to the right answer, if one exists, or detects linear inseparability in a finite number of steps. Since perceptrons divide the input space by a hyperplane, it is only reasonable to use them to learn rules which have approximately this structure: a small number of regions in input space separated by boundaries which are approximately hyperplanes. Watkin and Rau (1992b) have divided unlearnable rules into three categories: (i) rules in which there remains a unique **B** direction to be learned and the rule is monotonic in the overlap between examples and this direction (explained below), but there is a mismatch of thresholds; (ii) rules in which there remains a unique direction to be learned, but the rule is *not* monotonic in the overlap between examples and this direction; and (iii) rules in which there are several special directions in input space. For Ising perceptrons there is a further class of unlearnable problems: those in which the components of **B** vectors may take any real value, which is called mismatched weights (Seung, Sompolinsky, and Tishby, 1992). Since this problem is equivalent to an Ising perceptron learning to mimic a spherical perceptron, the problem is also called one of mismatched architecture.

An example of the first of these unlearnable rules is illustrated in Fig. 14(a), the learning of a linearly separable

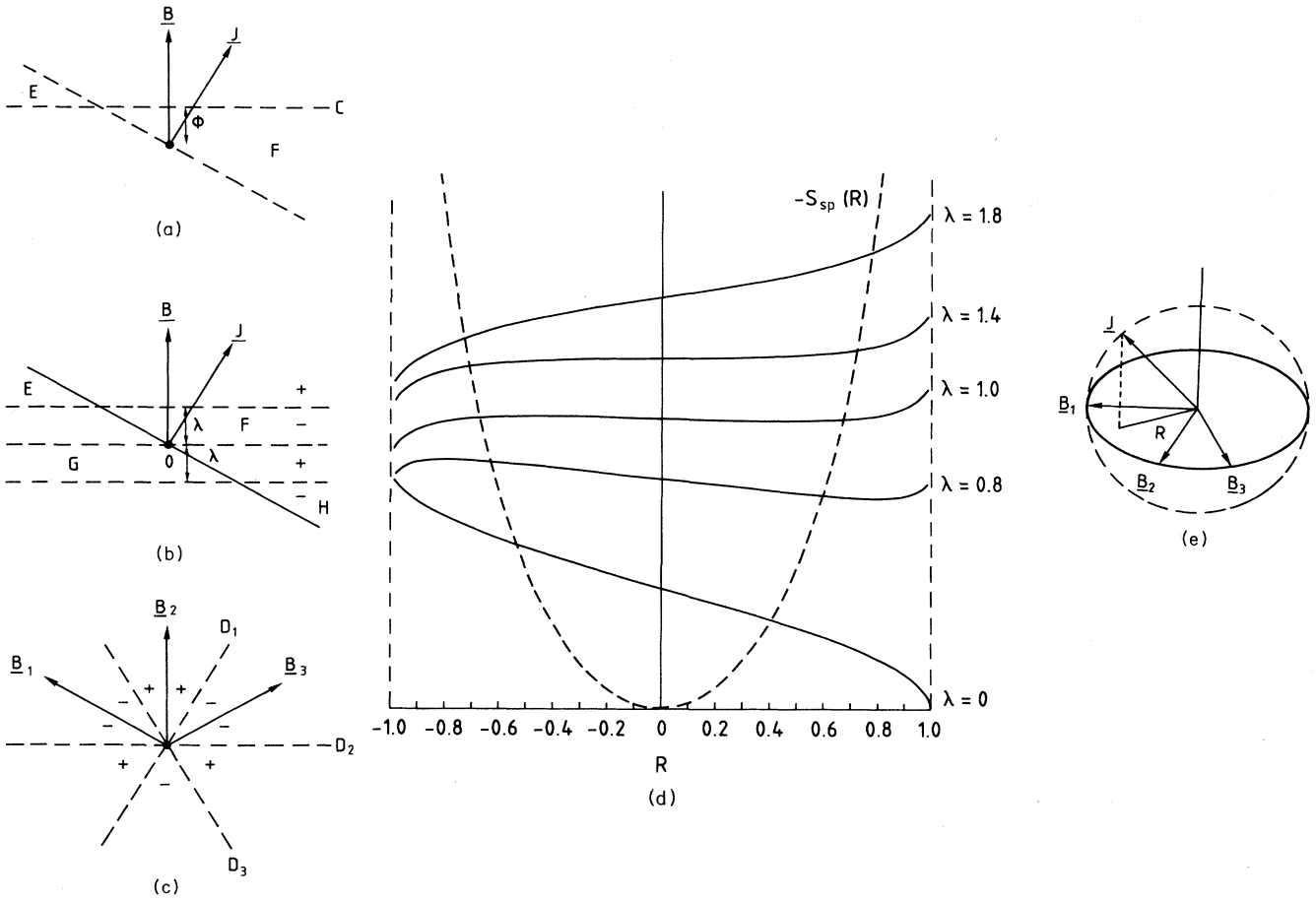

FIG. 14. Unlearnable rules: (a) The **B-J** plane, following Fig. 5(a), except that hyperplane $\mathcal{C}$, perpendicular to **B**, is displaced by ϕ from the origin; (b) The reversed-wedge problem. The correct answer to a question may be worked out by finding the position of the question's projection into the **B-J** plane and taking the sign marked in this diagram. **J** and **B** disagree on the answers to questions falling in areas *E*, *F*, *G*, and *H*. (c) The plane of the three teachers $\mathbf{B}_1$, $\mathbf{B}_2$, and $\mathbf{B}_3$, in the parity machine, perpendicular to the planes $\mathcal{D}_1$, $\mathcal{D}_2$, and $\mathcal{D}_3$, respectively. The signs in the diagram show the correct answers to questions. (d) Solid lines show the generalization function ϵ_f against R for five values of λ in the reversed-wedge problem. The dashed line shows $-S_{sp}$, minus the entropy for the spherical perceptron. The vertical scale is different for all lines. (e) The three-dimensional space containing **J** and the set of teacher vectors.

rule, Eq. (2.10), with a threshold ϕ, being learned by a perceptron with no threshold. Examples are answered $\mathcal{R}=+1$ if they fall on the positive **B** side of plane $\mathcal{C}$, which is normal to **B** and displaced by ϕ from the origin $\mathcal{O}$.

Figure 14(b) shows an example of the second class of unlearnable rule. A question **r** with a high overlap with **B**, $\mathbf{B}\cdot\mathbf{r}>\lambda/\sqrt{N}$, is answered $+1$, and one with $\mathbf{B}\cdot\mathbf{r}<-\lambda/\sqrt{N}$ is answered -1; but there exists a region centered on the origin where positive $\mathbf{B}\cdot\mathbf{r}$ gives a negative $\mathcal{R}$, and vice versa. We call this region a reversed wedge; the output $\mathcal{R}$ is not a monotonic function of $\mathbf{B}\cdot\mathbf{r}$.

The rule shown in Fig. 14(c) is one of the third class of problems,

$$V(\mathbf{r})=\mathrm{sgn}[(\mathbf{B}_1\cdot\mathbf{r})(\mathbf{B}_2\cdot\mathbf{r})\cdots(\mathbf{B}_k\cdot\mathbf{r})]\ , \tag{5.1}$$

where $\mathbf{B}_1,\mathbf{B}_2,\ldots,\mathbf{B}_k$ are k unit teacher vectors. Such a rule is called a parity machine (see Sec. VI.A). If the rule vectors are at right angles to each other (for $k>1$), or if $k\to\infty$, then the problem is *completely unlearnable* by a perceptron; every direction of **J** gives the same generalization function, $\epsilon_f(\mathbf{J},V)=\frac{1}{2}$.

Here we shall restrict k to be odd and the teacher vectors to be equally spaced out on a semicircle around the origin on a single two-dimensional plane, so that the angle between $\mathbf{B}_\gamma$ and $\mathbf{B}_{\gamma+1}$ is π/k for $1<\gamma<k-1$. This is illustrated for $k=3$ in Fig. 14(c). Examples falling into the "+" region are answered $+1$. Note that there is more than one special direction; in particular, $\mathbf{B}_1$ is equivalent to $\mathbf{B}_3$ and to $-\mathbf{B}_2$.

All these rules have been analyzed (Watkin and Rau, 1992b) at high temperature for Ising or spherical **J**, but at low temperature only for spherical **J**, due to the difficulties, explained in Secs. III and IV, of interpreting an Ising analysis.

1. A rule with a threshold

We consider first high-temperature learning (Sec. III.A.5). Since the first two categories of problems still have a unique direction to be learned, the entropy remains the same function of the only order parameter $R\equiv\mathbf{J}\cdot\mathbf{B}$, that is, Eq. (3.33) in the spherical case and Eq. (3.36) in the Ising case. Only $\epsilon_f(\mathbf{J},\mathbf{B})$, which is the proportion of examples falling into areas E and F in Fig. 14(a), is different. It can be easily calculated in a manner similar to that which gave Eq. (2.16), from the observation that the projection of random examples into any random direction has a Gaussian distribution given above by Eq. (2.18).

Learning at high temperatures with a spherical perceptron yields a smooth evolution towards the optimal result, although of course the minimum $\epsilon_f(\mathbf{J},\mathbf{B})$ is greater than zero. The Ising perceptron also finds the state with the minimum ϵ_g, as in Sec. III.C.1 although the first-order transition is not to **B** at once, but only into the region of **B**, that is, $R\approx 1$. $R\to 1$ only as $\tilde{\alpha}\to\infty$. Surprisingly, for thresholds greater than $\phi=0.22$, the first-order transition disappears.

The zero-temperature maximum stability result for a spherical perceptron also converges smoothly to the correct answer, although for some value of α (dependent upon ϕ) the value of $\kappa_{\max}$ changes sign. This is because, since not all examples can be perfectly learned, the perceptron must be allowed to get certain examples wrong before the energy function, Eq. (3.27), can be made zero. In other words, whereas in Fig. 8(b) taking κ to be positive forced the **J** into the center of the version space, here it must be allowed to become negative so that the space where $E_{\mathrm{MSA}}(\mathbf{J})$ is zero has nonzero volume: examples are allowed to lie $-\kappa$ outside the central region of Fig. 8(b). Note that for $\phi=\pm\infty$ answers are uncorrelated with questions, so that learning becomes equivalent to storing random input-output configurations (memories). This problem, mentioned in Sec. II, is equivalent to having no rule at all and has been fully studied in the neural network literature (for a review see, for example, Gutfreund and Toulouse, 1992). It is well known that a perceptron may store exactly $2N$ uncorrelated question-and-answer pairs (Winder, 1963; Cover, 1965; Venkatesh, 1986; Gardner, 1988), and thus as $\phi\to\pm\infty$, κ changes sign at $\alpha=2$

It was shown by Gardner and Derrida (1988) that replica symmetry breaking occurs in the storage of uncorrelated patterns when κ changes sign. This is not a result of frustration, since we have written our energy function so that all constraints on **J** can be satisfied; rather, the space of all **J** which make Eq. (3.27) zero becomes disconnected. The same result applies to learning the threshold problem for $\phi=\pm\infty$, where the sign change of κ itself implies that the problem is unlearnable. It has yet to be shown how great the effects of RSB are, but it rarely produces a qualitative difference in results. Nor is it yet known for general ϕ whether replica symmetry is always broken when κ changes sign. Might this be a general result of unlearnable problems?

2. The reversed-wedge problem

For $\lambda\to\infty$ and $\lambda=0$ the reversed-wedge problem becomes linearly separable, although for $\lambda\to\infty$ **J** should become aligned to $-\mathbf{B}$. For intermediate values, however, the perceptron could learn either the central region or the outer one. Watkin and Rau (1992b) worked out $\epsilon_f(\mathbf{J},\mathbf{B})$ by a calculation very similar to that for the threshold problem, and the result is plotted in Fig. 14(d) for five values of λ. It is easy to see why the minimum at positive R moves away from $R=1$ as λ rises from zero, since, considering Fig. 14(b), we see that the perceptron answers questions in regions E, F, G, and H wrongly. For $\lambda\neq 0$ we can make large reductions in F and G by making R a little less than 1, only increasing regions E and H, which are a long way from the origin and where few examples fall.

The dashed line in Fig. 14(d) shows the value of $-s_{\mathrm{sp}}(R)$ for spherical perceptrons [Eq. (3.33)]. Clearly

for small $\tilde{\alpha}$, where the high-temperature results are dominated by the entropy, minima with small $|R|$ are preferred, which for intermediate λ means the one at positive R, even though this is not the global minimum of ϵ_g for $\lambda > 0.818$. Once we have reached it a higher number of examples will not help us to escape, because there is an infinite energy barrier to overcome to reach $R = -1$ (as $N \to \infty$). Thus $\mathbf{J}$ is trapped in a spurious minimum. It is better to begin learning after a finite value of α has been reached, so that the free-energy surface guides the perceptron at once to the global minimum of ϵ_g.

Zero-temperature learning is even more problematic. The MSA, as explained in Sec. III.B.1, finds the $\mathbf{J}$ which gets its *worst* example most right (or least wrong), in the sense of making $\xi_0^\mu \mathbf{J} \cdot \xi^\mu$ most positive (or least negative). As $\alpha \to \infty$ there will be many examples falling in the outer region for any value of λ; a $\mathbf{J}$ which learns the outer region will get only those examples which fall in the inner region wrong, and so it will have $\kappa \geq -\lambda$, while a $\mathbf{J}$ learning the inner region will get all the outer examples wrong and so have $\kappa \leq -\lambda$. Thus the maximum stability rule *always* learns the outer region perfectly: $R \to 1$ as $\alpha \to \infty$. Since we see from Fig. 14(d) that for $\lambda \neq 0$ the minimum of ϵ_f is at $R < 1$, this implies an *overlearning* of direction $\mathbf{B}$. For large λ, $\epsilon_g(\alpha) > \frac{1}{2}$ for high α, which is worse than a random choice for $\mathbf{J}$. It seems that in unlearnable problems the MSA is the wrong one to use.

A possible solution is to keep $\kappa = 0$ and just minimize E_t, the number of wrongly classified examples by the techniques of Sec. III. The numerical analysis of the saddle-point equations is difficult, but it seems likely that no overlearning is present and that the system converges smoothly to the best possible solution. However, it seems likely as well that there will be replica symmetry breaking, because for high α, the minimum $E_t > 0$ which implies that all the constraints on $\mathbf{J}$ cannot be met and there is frustration, which in spin glasses is commonly associated with RSB, and the presence of many metastable states which are a hazard for any learning algorithm.

Of course prior information about the form of the rule gives us much better learning algorithms. Watkin (1991) has shown that for some values of λ, optimal learning with a spherical perceptron (taking into account the form of the rule) gives a $\mathbf{J}$ with the minimum generalization error for finite α, if $\lambda \neq 0$. This is in contrast even to reliable high-temperature learning, for which ϵ_g tends to its minimum only as $\tilde{\alpha} \to \infty$.

3. The parity machine

In Sec. II.D, ϵ_f was calculated for a perceptron learning linearly separable problems using the lengths of arcs on the unit circle in the two-dimensional space containing $\mathbf{J}$ and $\mathbf{B}$. Entropy was calculated in Sec. III.C.1 from the intersection of the circle with a line. For the parity machine, by analogy we require a sketch of the three-dimensional space containing $\mathbf{J}$ and the plane of the set $\{\mathbf{B}^\gamma\}$, Fig. 14(e). $\epsilon_f(\mathbf{J}, V)$ is calculated (Watkin and Rau, 1992b) from areas of spherical triangles on the surface of the unit sphere, and entropy from the intersection of the sphere with a plane; it turns out that the contribution to the entropy of the polar angle of $\mathbf{J}$ in the $\mathbf{B}$-$\mathbf{J}$ space does not scale extensively and may be ignored. Thus for high-T learning, the only order parameter is the projection of $\mathbf{J}$ into the plane of the $\{\mathbf{B}^\gamma\}$, R, and for any R the polar angle will automatically be such as to minimize $\epsilon_f(\mathbf{J}, V)$.

Although $\epsilon_f(\mathbf{J}, V)$ decreases smoothly from $R = 0$ to $R = 1$, the free energy f always has two minima, one at $R = 0$ and another at a value of R, which tends to 1 as $\tilde{\alpha} \to \infty$. Even with unlimited examples, a Monte Carlo algorithm will not be able to overcome the energy barrier between the minima—even though it becomes very small as $\tilde{\alpha} \to \infty$. To converge to the answer with minimum $\epsilon_f(\mathbf{J}, V)$, we must begin the simulation inside the basin of attraction of the high-R solution, which means that we require some prior information about the teacher vectors, no matter how many examples we have (or else we must rely on finite-size effects; see Sec. VI.C.1).

Maximum stability fails again in this situation; again it finds the $\mathbf{J}$ with the *largest* generalization error. Simply minimizing $E_t(\mathbf{J})$ is liable to result in the same problems as high-temperature learning, as well as the problems which the technique encountered in the reversed-wedge problem.

Hebb learning, Eq. (2.17) in Sec. II.D fails completely for this rule, since, because of the symmetry of the teacher plane, there will be no constructive component of the sum over examples in any direction. Optimal learning of this rule should be straightforward, but has not so far been analyzed.

The main conclusion (Watkin and Rau, 1992b) is that the MSA is very unreliable for unlearnable problems, since the $\mathbf{J}$ it constructs is determined by examples which cannot be learned, not from those which can. Minimizing E_t is little better, since the frustration it necessarily produces probably causes replica symmetry breaking. In general, without prior information the high-temperature algorithm is most suitable if sufficient examples are available, but judicious use of prior information may significantly enhance performance.

4. Binary perceptron learning with mismatched weights

The final type of unlearnable rule is the case in which the gauge constraints on the weight space and rule space are different. Seung, Sompolinsky, and Tishby (1992) have analyzed the case of an Ising perceptron learning a spherical perceptron (we shall report only the results for perceptrons with binary outputs). Each of the components of $\mathbf{B}$ is drawn from a Gaussian distribution of width 1 and mean zero, which means that on average $\mathbf{B}$ is constrained to lie on the N-dimensional unit sphere. The student's weights are taken to be $\{J_i = \pm 1\}$, so that $\mathbf{J}$ is one of the vertices of an N-dimensional hypercube and there is an error between every component of the teacher

and the student. Thus the minimum generalization function is larger than zero, $\epsilon_f(\mathbf{J},\mathbf{B}) \geq \epsilon_{\min} = (1/\pi) \cos^{-1}(\sqrt{2/\pi}) \approx 0.21$.

For any fixed temperature, minimizing E_t yields, according to RS theory, an asymptotic dependence of ϵ_g on the number of examples as $\epsilon_g - \epsilon_{\min} \sim 1/\sqrt{\alpha}$, which is as slow as Hebb and pseudoinverse learning of learnable rules (described in Secs. II.D and III.C.1, respectively). In this instance the annealed approximation fails completely, predicting $\epsilon_g - \epsilon_{\min} \sim \alpha^{-2}$.

Below $T \approx 1$, numerical simulations show substantial deviations of the training error from the quenched result for values of α below the zero-entropy line of the RS theory, where RS should be exact (although this may be due to problems in properly equilibrating the system). On the other hand, the generalization error in simulations does agree with the theory (at least for training temperatures greater than zero).

As explained in Sec. II.G, minimizing the training error is not always optimal if we have some prior knowledge about the rule; and, in fact, in this case, if E_t is used as the energy, ϵ_g does not have a monotonic dependence upon training temperature. Seung, Sompolinsky, and Tishby (1992) showed that the best temperature is zero below $\alpha \approx 1.27$ and increases above this value as $T_{\text{opt}}(\alpha) \sim \alpha^{3/5}$ for large α. If one employs the best temperature at each value of α, then learning becomes asymptotically faster, $\epsilon_{\min} - \epsilon_g \sim \alpha^{-4/5}$ turns out to be true at any low temperature. The zero-temperature prediction for generalization error is shown as line 1 in Fig. 15. The zero-temperature prediction for training error is line 2.

Watkin (1993) analyzed optimal learning of the problem. Since the rule space is the same in this problem as in the problem in Sec. III.B.2, the network error $\epsilon_n(\mathbf{J})$ is given by the same formula (3.30), and identity (3.31) also applies. $\mathbf{J}^{\text{opt}}$ is therefore the Ising vector with the maximal overlap with $\sum_r \mathbf{B}^r$, where the $\{\mathbf{B}^r\}$ are again random *spherical* samplers in the version space of spherical rules. Thus $J_i^{\text{opt}} = \text{sgn}(\sum_r B_i^r)$. The sharp peak in the distribution of the overlap of samplers [at $\mathbf{B}^r \cdot \mathbf{B}^{r'} = q(\alpha)$ for $r \neq r'$, where $q(\alpha)$ is the overlap of two points in a version space of spherical vectors, as calculated in Sec. III.A] easily gives the optimal generalization ability $\epsilon_g = (1/\pi) \cos^{-1}([2q(\alpha)/\pi]^{1/2})$, which has the asymptotic behavior $\epsilon_n(\mathbf{J}^{\text{opt}}) - \epsilon_g^{\min} \sim \alpha^{-2}$ as $\alpha \to \infty$. Simulations are very easy, since they just involve "clipping" the optimal spherical $\mathbf{J}$ obtained in Sec. III.C.1. The results are shown in Fig. 15: the solid line 3 is the theoretical prediction, and the means and error bars are from 200 independent runs using $N = 50$ and 25 samplers.

Modern integrated circuits are invariably designed by computers of much greater power. It is therefore not unrealistic in advanced applications to consider a learning process in which inference is performed assuming a rule space which is continuous. A network of discrete couplings could then be constructed to agree with samples of the continuous space. Thus construction of the network would be performed in a discrete space in which the dependence on the disordered training set has been

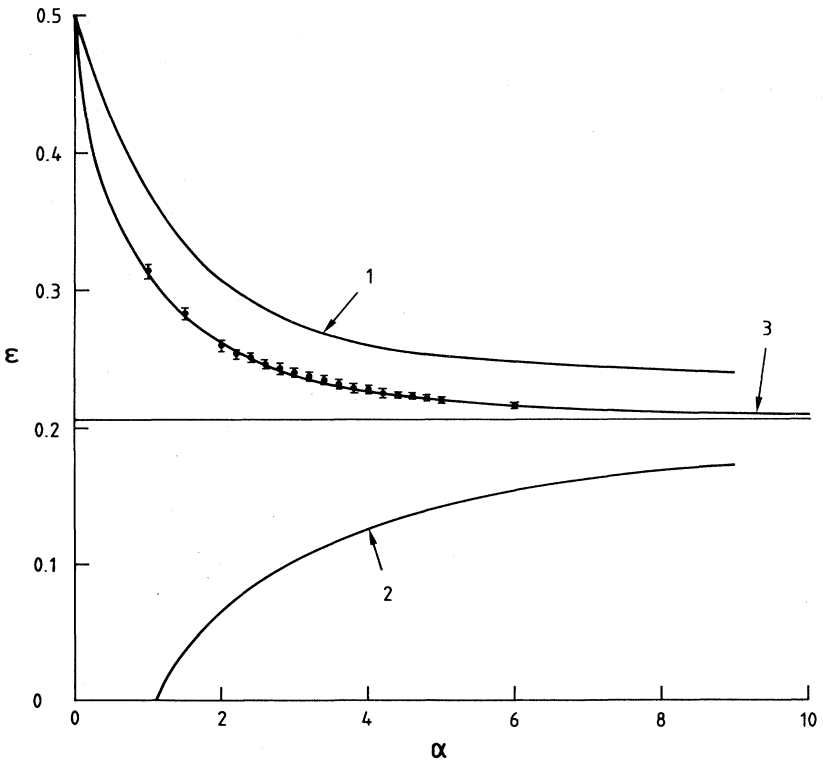

FIG. 15. Learning the mismatched weight problem. Lines 1 and 2 show ϵ_g and ϵ_t for stochastic minimization of E_t. Line 3 shows ϵ_g for optimal learning, with means and error bars from 200 independent simulations using $N = 50$ and $m = 25$ (Watkin, 1993).

smoothed out. If, as seems generally true, discrete spaces are much more prone than continuous ones to ergodicity breaking in the presence of quenched disorder, then dividing the construction of a network into these two steps may be very efficient in computer time, as well as for giving better generalization.

B. Noisy examples

Elsewhere in this paper we work with noiseless examples: perfect data produced by the underlying rule. In many applications, however, the data will be corrupted, and here we briefly describe what is a straightforward extension to the theory.

Noise in the examples has been formulated in two ways. In the first (Györgyi and Tishby, 1990) the *questions* of the training set are corrupted: a set of questions $\{\xi^\mu\}$ was used to produce a set of correct answers $\{\xi_o^\mu\}$, and we learn from $\{\xi_o^\mu\}$ and a corrupted version of the questions. An alternative way to introduce noise assumes that *answers* are corrupted: we are given a set of questions $\{\xi^\mu\}$ and answers $\{\xi_o^\mu\}$ of which a random fraction χ are corrupted, i.e., $\xi_o^\mu \neq V(\xi^\mu)$ for a random fraction χ of the examples (Opper and Haussler, 1991b).

The first case was studied by Györgyi and Tishby (1990) under the approach of Sec. III.A.2: a spherical binary perceptron learns a linearly separable rule by a stochastic process in the space of networks on a landscape given by E_t at temperature T. In contrast to the noiseless case, the following effects were found

(i) RSB occurs at low temperatures. This is because noisy examples lead to incompatible constraints on **J**, and hence to frustration.

(ii) For a given level of noise, there is a value of α above which the generalization error is not a monotonic function of T. It can be better to use a nonzero temperature (to avoid overfitting incorrect data).

The same effects are observed, too, if answers are corrupted instead of questions (Penney and Wendemuth, private communication, 1992).

Opper and Haussler (1991b) considered noisy examples within the Bayesian approach to learning. If a random fraction χ of the examples are corrupted, then $P(\xi_o^\mu | V, \xi^\mu)$, the probability that question ξ^μ is answered by ξ_o^μ given that the underlying rule is V, is $(1-\chi)$ if $\xi_o^\mu = V(\xi^\mu)$ and χ if $\xi_o^\mu \neq V(\xi^\mu)$. Thus, given a training set, we can calculate the posterior probability of the underlying rule being V as

$$P(V/\{\xi^\mu,\xi_o^\mu\}) = \frac{\prod_\mu P(\xi_o^\mu/V,\xi^\mu)P(V)}{\int dV \prod_\mu P(\xi_o^\mu/V,\xi^\mu)P(V)} , \quad (5.2)$$

where

$$P(\xi_o^\mu/V,\xi^\mu) = (1-\chi)\delta(V(\xi^\mu),\xi_o^\mu) + \chi[1-\delta(V(\xi^\mu),\xi_o^\mu)] .$$

But this can be rewritten as

$$P(V/\{\xi^\mu,\xi_o^\mu\}) = \frac{e^{-\beta E_t(V,\{\xi^\mu,\xi_o^\mu\})}P(V)}{Z} , \quad (5.3)$$

where E_t is the training error, defined by Eq. (2.11),

$$Z = \int dV\, e^{-\beta E_t(V,\{\xi^\mu,\xi_o^\mu\})}P(V) , \quad (5.4)$$

and $\beta = \ln[(1-\chi)/\chi]$ (Opper and Haussler, 1991b). That is, our knowledge of the rule is described by a partition function, with a Gibbs distribution over rule space at a *natural* temperature.

All the Bayesian machinery developed in Secs. II.F and II.G remain valid for this generalized situation, and it is easy to show the following:

(i) the generalization ability of the Bayes algorithm for a given number of examples falls monotonously as χ rises to $\frac{1}{2}$ (Opper and Haussler, 1991b). In fact, Eq. (2.31) remains valid for all χ, where q is the (self-averaging) overlap of two rules selected from the posterior probability distribution.

(ii) in the large N limit, the optimally taught binary perceptron generalizes as well as the Bayes algorithm for all χ. In this case, the samplers (Sec. III.B.2) are produced by a stochastic process at the natural temperature β given above.

C. Selected examples

1. A simple selection procedure for the perceptron

Elsewhere in this paper the questions in the training set are chosen randomly and independently, but a mechanism has been proposed (Kinzel and Ruján, 1990; Baum, 1990) in which the distribution of new examples depends upon what has already been learned. This interrogative approach to learning may have wider applications in the extraction of information from experiments. We noted above that the plane $\mathcal{D}_5$ in Fig. 4(c) places no new constraint upon **B**. The smaller the version space becomes, the fewer the examples which will give any new information about **B**. From Fig. 4(b) we can see that to constrain **B** in a small region (i.e., to minimize the version space), we would like examples to fall close to plane $\mathcal{C}$. These examples carry greater information, as can be seen from Fig. 16(a), which shows a schematic "top view" in the manner of Fig. 4(c). The plane $\mathcal{D}$ is perpendicular to an example (not marked) which is perpendicular to **B**. This plane must therefore divide the version space into two.

Of course, we do not know **B** to begin with, but if after p examples we have by some algorithm generated a best guess for **B** of $\mathbf{J}^{(p)}$, then we could *choose* the next question ξ^{p+1} to learn from such that

$$\sqrt{N}\,\mathbf{J}^{(p)}\cdot\xi^{p+1} = 0 \quad (5.5)$$

to first order in N.

We define $\theta^{(p)}$ as the angle between $\mathbf{J}^{(p)}$ and **B**, from $\cos(\theta^{(p)}) \equiv \mathbf{J}^{(p)}\cdot\mathbf{B}$, and observe that since the component

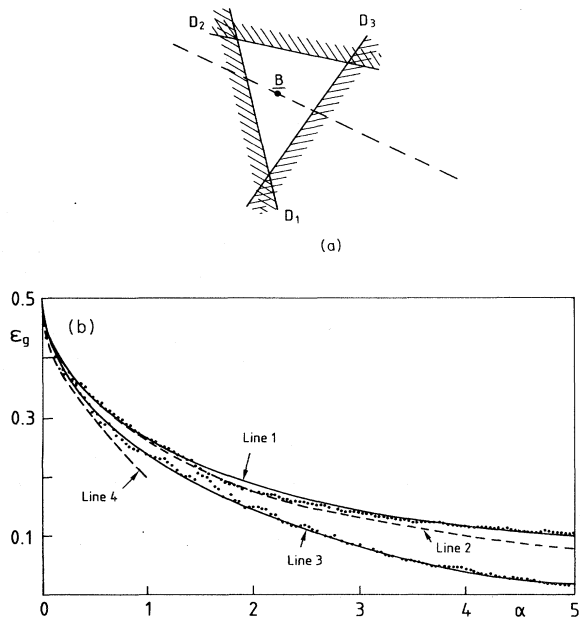

FIG. 16. Selecting examples: (a) A schematic diagram, following Fig. 4(c), shows **B** constrained to the version space by planes $\mathcal{D}_1$–$\mathcal{D}_3$, and the projection of hyperplane $\mathcal{D}$ which is perpendicular to an example (not marked) perpendicular to **B**. (b) Generalization error ϵ_g against α. Line 1 shows the results using Hebb learning, with data points from a simulation using $N=50$ (Kinzel and Ruján, 1990). Line 2 is a reminder of the Bayesian optimal result using unselected examples. Line 3 is the MSA result with selected examples, compared to data points from a simulation (Kinzel and Rujá, 1990), as explained in the text. Line 4 shows the result of choosing new questions to be perpendicular to all previous ones (for $\alpha<1$) and learning with the MSA.

of **B** perpendicular to **J** is $\sin(\theta^{(p)})$, the variable $x^{(p+1)}=\sqrt{N}\,\xi^{p+1}\cdot\mathbf{B}$ is normally distributed with

$$\text{Prob}(x^{(p+1)})=\frac{1}{\sqrt{2\pi}\sin(\theta^{(p)})}\exp\left\{-\frac{x^{(p+1)2}}{2\sin^2(\theta^{(p)})}\right\}. \tag{5.6}$$

If such questions are used in the Hebb algorithm, Eq. (2.17), it is straightforward to derive a recursion relation for $\theta^{(p+1)}$ in terms of $\theta^{(p)}$ and thus a differential equation for $\epsilon_g(\alpha)$. This argument assumes that only correlations between the $\{\xi^\mu\}$ in the direction of **B** are of significance, so the average over $\xi^{\mu+1}$ with constraint Eq. (5.5) is equivalent to the average with constraint Eq. (5.6), despite the fact that Eq. (5.6) does not itself imply Eq. (5.5). This ansatz, made explicit by Watkin and Rau (1992a), is very natural, however, because the hyperplane perpendicular to **B** is an $(N-1)$-dimensional subspace in which the component of **J** perpendicular to **B** varies widely as p increases. Similarly, the hyperplane perpendicular to $\mathbf{J}^{(p)}$, from which ξ^{p+1} is randomly chosen, is $(N-1)$ dimensional. The solution is plotted as line 1 in Fig. 16(b), with the average results of five runs of a simulation using $N=50$ shown dotted for comparison (Kinzel and Ruján, 1990). The curve is lower than the Hebb result for unselected examples, line 1 on Fig. 5(b) in Sec. II.D, but still higher than the optimal perceptron for unselected examples, Sec. III.C.1, which, as a remainder, is shown dashed as line 2 on Fig. 16(b).

Of course, what we would like to do is apply the same procedure to perceptrons taught with a more sophisticated rule, and numerically, using the MSA, this does indeed seem to be much better (Kinzel and Ruján, 1990). Simulations suggest, in fact, that the decay of ϵ_g is exponential in α, as is intuitively reasonable: from Fig. 16(a) each new example reduces the volume of the space by a factor which might be expected to become constant for high α. An exact analysis of the situation is somewhat difficult, however, because it amounts to solving a dynamical process. We must follow **J** on its journey towards **B** to find out what correlations exist between the examples generated in the process. The ansatz pointed out above, however, converts the problem back to one of statics.

Watkin and Rau (1992a) thus derived saddle-point equations similar to the solution with unselected examples (Opper *et al.*, 1990), and the difference has an appealing physical interpretation. A term which had been recognized as a Gaussian noise decreases in width as α increases (instead of remaining of constant width as for unselected examples), which corresponds to placing a more stringent constraint per example on **B** as α increases.

The numerical solution of the equations, line 3 in Fig. 16(b), agrees with the average results of seven runs of a numerical simulation using $N=50$, which are also shown dotted, to within the accuracy of both.

2. More general selection techniques

Recently, Seung, Opper, and Sompolinsky (1992) have proposed a more general method of selecting examples. They observe that the value of a new question can be determined by considering how samplers (see Sec. III.B.2) placed randomly inside the version space answer it: a question on whose answer many samplers disagree is a good one to ask. Thus the "query-by-committee" algorithm selects a question which half the samplers in version space answer by $+1$ and the other half by -1. As the number of samplers rises to infinity, every selected example divides the version space into two equal halves, yielding the maximal information gain of 1 bit per question.

Seung, Opper, and Sompolinsky (1992) pointed out that the Shannon information gain of a new example is a helpful tool for understanding the efficiency of selecting-examples algorithms. Here the entropy per site s is $(1/N)\ln(\text{Vol})$, where Vol is the volume of the version space. For example, if a linearly separable rule defined by a spherical **B** is learned by a spherical binary perceptron using *random* questions, then

$$s \sim \ln(\epsilon_g) \tag{5.7}$$

for large α. The information gain per question, defined as $I(\alpha) \equiv -\partial s / \partial \alpha$, can be shown to be proportional to ϵ_g. As α rises, $I(\alpha)$ falls to zero, agreeing with the observation that random examples provide less and less new information.

A statistical-mechanics analysis of the query-by-committee algorithm is subject to the same technical difficulties described in Sec. V.C.1: a dynamical process has to be solved in which "time" corresponds to the number of examples. Seung, Opper, and Sompolinsky (1992) use the replica method and make the assumption that the typical overlap of two perceptron vectors at different times t and t' is equal to the typical overlap of two vectors at the earlier time: $\mathbf{J}(t)\cdot\mathbf{J}(t') = q(\min\{t,t'\})$. Using this approximation they can show that the asymptotic information gain $I(\infty)$ for $\alpha \to \infty$ is finite. Assuming the validity of Eq. (5.7) gives

$$\epsilon_g \sim e^{-\alpha I(\infty)} . \tag{5.8}$$

The learning curve obtained lies slightly below line 3 in Fig. 16(b), and numerical simulations agree well with these predictions.

One drawback of the query-by-committee algorithm is that the time needed to select a question providing maximum disagreement diverges as $1/\epsilon_g$. Simple, direct algorithms, such as the one of Kinzel and Ruján (1990), are much faster, though they are not so easy to generalize to multilayer networks as query-by-committee.

Incidentally, we can show that selection of examples in accordance with Eq. (5.5) or with the query-by-committee algorithm is not optimal by suggesting the following scheme for the perceptron: we insist that examples be chosen to divide version space into equal halves which should also be "as convex as possible," with the justification that, since the rule $\mathbf{B}$ lies randomly in one half, its average overlap with the other points in that half would then be maximal; the student $\mathbf{J}$ should then be set using the maximum stability algorithm. For $\alpha < 1$ this manner of selecting examples is equivalent to insisting that new questions be perpendicular to all previous questions. The generalization error may therefore be calculated geometrically $[\epsilon_g(\alpha) = \cos^{-1}(\sqrt{2\alpha/\pi})/\pi]$ and is marked as line 4 on Fig. 16(b), which lies below line 3 and also below the query-by-committee result. In fact, for $\alpha = 1$, it lies as far below line 3 as line 3 lies below the maximum stability result for unselected examples (line 2 on Fig. 10). This manner of selection of examples is probably optimal for the perceptron, but its consequences are trickier to calculate for $\alpha > 1$and so far have not been analyzed. It is also not clear how it might be generalized to the selection of examples for more complex networks.

D. Online learning

This paper has developed an information-theoretic approach to learning in which the best use is made of the training set. In many problems, however, the number of examples is practically unlimited, so that our primary interest is in minimizing the computational effort of the learning algorithm.

One such approach is online learning or real-time learning, which is computationally cheap and even does without storage of the training set. Suppose that the examples we are learning from are presented in a sequence. Online learning means that at every moment the network is adapted simply in response to the newest example, which is then forgotten.

Clearly, Hebb learning, which was discussed in Sec. II.D, may be regarded as online learning, since,

$$\mathbf{J}^{(p+1)} = \mathbf{J}^{(p)} + \frac{1}{N} \xi_o^{p+1} \boldsymbol{\xi}^{p+1} \tag{5.9}$$

(disregarding the normalization constant γ). Here we shall briefly mention two other interesting examples of online learning

1. A modified Hebb algorithm

Kinouchi and Caticha (1992) presented a modified version of the Hebb algorithm for a binary spherical perceptron. Equation (5.9) is generalized to

$$\mathbf{J}^{(p+1)} = \mathbf{J}^{(p)} + \frac{1}{N} W(x_B^{(p+1)}, x_j^{(p+1)}) \boldsymbol{\xi}^{p+1} \xi_o^{p+1} , \tag{5.10}$$

where $x_B^{(p+1)} \equiv \sqrt{N}\, \mathbf{B}\cdot\boldsymbol{\xi}^{p+1}$ and $x_J^{(p+1)} \equiv \sqrt{N}\, \mathbf{J}\cdot\boldsymbol{\xi}^{p+1}$. We can now calculate the success of learning in a way similar to that used in Sec. II.D. The result is a formula for the generalization error as a function of α and as a functional of W. They then optimized with respect to W to find the optimal $W^*(x_B, x_J)$. Of course, in a real experiment $x_B^{(p)}$ is not known—only $\mathrm{sgn}(x_B^{(p)}) = \xi_o^{p+1}$ is given. Kinouchi and Caticha (1992) therefore suggest using

$$\mathbf{J}^{(p+1)} = \mathbf{J}^{(p)} + \frac{1}{N} \overline{W} \xi_o^{p+1} \boldsymbol{\xi}^{p+1} , \tag{5.11}$$

where $\overline{W}$ is W^* averaged over the posterior probability density of $x_B^{(p)}$ given just ξ_o^{p+1} and $x_J^{(p)}$,

$$\overline{W} \equiv \int dx_B^{(p)} W^*(x_B^{(p)}, x_J^{(p)}) P^{\mathrm{post}}(x_B^{(p)} | \xi_o^{p+1}, x_J^{(p)}) . \tag{5.12}$$

They found a generalization error significantly smaller than with the Hebb algorithm.

Departing from the philosophy of online learning, Kinouchi and Caticha (1992) then considered learning the same set of examples again using Eq. (5.10). Doing this several times, they found numerically that generalization approaches the optimal learning result (Sec. III.C.1.a). This algorithm is faster than the method of samplers, but requires a knowledge of the distribution of questions to find $\overline{W}$. It is also not clear how to extend the modified Hebb algorithm to more complicated networks.

Kinouchi and Caticha (1992) also applied this formalism to learning from examples which have been selected in a way such that $\sqrt{N}\, \mathbf{J}^{(p)}\cdot\boldsymbol{\xi}^{p+1} = 0$, as in Sec. V.C.1. In

this case the optimal W yields a generalization error

$$\epsilon_g \sim \frac{1}{\pi} e^{-\alpha/\pi}, \quad \alpha \gg 1 , \tag{5.13}$$

and $\overline{W}$ is just $\sqrt{2/\pi}\exp(-\alpha/\pi)$. This is better than any known algorithm using random examples and is computationally cheaper than the MSA procedure suggested by Kinzel and Ruján (1991).

2. A time-varying rule

One can conceive of a rule to be learned which is not static. For example, the rules governing the evolution of the stock market are subject to changing legal restrictions or technological advances. Within the VC theory (Sec. II.E), Helmbold and Long (1991) studied the "tracking of drifting concepts using random examples" on an abstract level independent of the specific form of the rule, and derived rigorous bounds on the generalization error that can be achieved, given a certain amount of drift.

Recently Biehl and Schwarze (1992) considered learning a time-dependent linearly separable rule, in which the teacher vector $\mathbf{B}$ evolves by a random walk (on the unit N-sphere), so that if we call $\mathbf{B}^{(p)}$ the rule after the training set contains p examples, then $\mathbf{B}^{(p)}\cdot\mathbf{B}^{(p+1)}=1-(\eta/N)$. Recent examples of the rule give more information than older ones, so the learning rule must weight recent examples more strongly. This can be achieved by introducing weight decay into the learning rule,

$$\mathbf{J}^{(p+1)}=\left[1-\frac{\lambda}{N}\right]\mathbf{J}^{(p)}+\frac{1}{N}W(x_B^{(p+1)},x_J^{(p+1)})\xi_o^{p+1}\xi^{p+1} . \tag{5.14}$$

Weight decay had already been studied in the context of attractor neural networks working as "forgetful memories" (Mézard, Nadal, and Toulouse, 1986; Derrida and Nadal, 1987; van Hemmen, Keller, and Kühn, 1988; Biehl, 1989; Kinzel and Opper, 1991). Biehl and Schwarze (1992) found that no online scheme of type (5.14) could keep track of the evolving rule *perfectly* for nonzero η.

E. Multiclass classification

Practical learning problems to which neural networks have been applied generally have many more than two possible answers to questions, and the components of the questions, too, may naturally take more than two values. Many scientific applications exist, such as the classification of phonemes (Kohonen, 1988), or recognition of text (Schmitz *et al.*, 1990), or the prediction of the secondary structure of proteins from the local sequence of amino acids (Bohr *et al.*, 1990).

Of course many values can be encoded as combinations of binary outputs, but this introduces an unwelcome arbitrariness in the output representation and introduces relationships between answers which will not in general exist. It also requires more neurons, connections, etc. We would rather each output neuron performed a more complicated function of the states of its input nodes.

Recently Watkin, Rau, Bollé, and van Mourik (1992) suggested that since, in general, the Q' possible answers to questions will be equivalent (which is the natural assumption without *a priori* knowledge), the natural solution is a perceptron with an output node whose Q' output states obey the "Potts symmetry," familiar from equilibrium statistical mechanics (for a review see Wu, 1982). The states are equivalent, like the Q' vertices of a $(Q'-1)$-dimensional tetrahedron. The states of the input nodes, too, take Q equivalent values. This system is called a *multiclass* perceptron or a *Potts* perceptron. Potts neurons have been used before (Kanter, 1988; Bollé and Dupont, 1990; Bollé *et al.*, 1991; Bollé *et al.*, 1992) in the very different problem of storing patterns in a highly recurrent neural network, which implies that $Q=Q'$. Their structure, and in particular their remarkable gauge invariances, were investigated by Nadal and Rau (1991).

Watkin, Rau, Bollé, and van Mourik (1992) analyzed Hebb learning for the Potts perceptron by considering which components scaled constructively and which did not, the same arguments which were discussed in Sec. II.D for the binary perceptron. This is a great deal easier than a conventional statistical-mechanics formulation, which rapidly becomes very involved. The analysis may be further simplified by exploiting the gauge invariances of the Potts perceptron, giving smooth convergence to the perfect answer, with $\epsilon_g \sim 1/\sqrt{\alpha}$, as in the binary case.

It seems, in fact, that, reassuringly, the main results of binary perceptron learning will be preserved in their multiclass analogs. Multiclass analogs of the Ising perceptron, for example, show sudden transitions to perfect generalization for high-temperature learning. However, no zero-temperature analogs of maximum stability algorithms have so far been investigated, and it is well known that frustrated Potts systems show quite different low-temperature behavior from Ising ones, with, for example, quite different schemes of RSB (Sherrington, 1986). We shall briefly meet multiclass perceptrons again in Sec. V.F.

It should be pointed out that there do exist rules for which the answers are in some way structured, and with this prior knowledge a choice of perceptron whose answers also possess this structure might be even more efficient. In quality control, for example, the answers might have a ladder structure with levels of quality; for these the output node should be a graded response neuron, whose states have a definite order (Kühn *et al.*, 1991).

F. The proximity problem

A fundamentally different form of rule was studied by Del Giudice *et al.* (1989) and by Hansel and Sompolinsky (1990), the *proximity* or *prototype* problem. Instead

of a teacher **B**, we begin with p_0 random, uncorrelated N-vector prototypes $\{\boldsymbol{\eta}^\mu\}$, $\mu=1,\ldots,p_0$, each of which has a random Ising output $\{\eta_o^\mu\}$. Hansel and Sompolinsky (1990) studied the interesting case of an extensive number of prototypes, $p_0 \propto N$. The correct answer for any input **S** is the correct output of the prototype closest in Hamming distance to **S**. The rule is learned from p examples of each prototype $\{\boldsymbol{\xi}^{\mu l}\}$, $l=1,\ldots,p$, chosen at random but with the constraint that $\boldsymbol{\eta}^\mu\cdot\boldsymbol{\xi}^{\mu l}=m$. For an extensive p_o this problem is clearly unlearnable by a perceptron, since no plane can divide the input space to correctly answer every possible input; so this problem is a good model for rules very much more complicated than linearly separable ones. Hansel and Sompolinsky (1990) minimized the training error $E_t(\mathbf{J})$ within the student space through the techniques of Sec. III.A and considered the limit of m small, which implies large p, since p must be rescaled as $\tilde{p}=m^2p/(1-m^2)$, which remains of order 1. For $\tilde{p}$ less than a critical value $\tilde{p}_c$, a **J** may be found which makes the training energy zero; in this range the consequent generalization error [marked as line 1 in Fig. 17, which shows the example of $p_0=1.6Nm^2/(1-m^2)$] falls from $\frac{1}{2}$ as $\tilde{p}$ rises, but then climbs slightly as $\tilde{p}\to\tilde{p}_c$, since the only **J** which correctly learns all examples has overfitting (see Sec. III.C.1). For $\tilde{p}>\tilde{p}_c$, the training error rises smoothly (line 2) and the generalization error falls; both tend to the same value $\epsilon_{\min}(m,p_0)$ as $\tilde{p}\to\infty$. Training with a finite training error (equivalent to a finite temperature) eliminates the problem of overfitting, and the asymptotic behavior of the generalization error is $\epsilon_g-\epsilon_{\min}\sim\tilde{p}^{-1}$, as shown in $\epsilon_g-\epsilon_{\min}\sim\tilde{p}^{-1}$, as shown in Fig. 17, line 3.

It is possible to map this learning problem to a different one previously studied in the neural network literature (Wong and Sherrington, 1990), where a large network with very few random connections is taught to

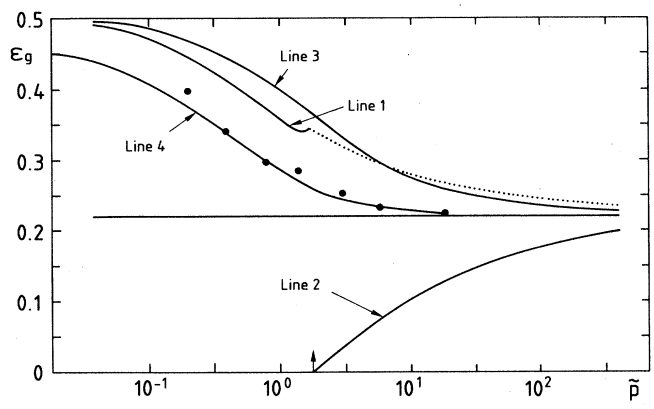

FIG. 17. Error against $\tilde{p}$ in the proximity problem using $p_0=1.6Nm^2(1-m^2)$, after Hansel and Sompolinsky (1990) and Watkin, Rau, Bollé, and van Mourik (1992). Line 1 shows the ϵ_g found by simply minimizing E_t; ϵ_g tends to 0.21 as $\tilde{p}\to\infty$. Line 2 is the respective training error. Line 3 shows the generalization error if the training error is fixed at 0.21. Line 4 shows the optimal Hebb algorithm, compared to a single simulation using $N=5000$ and $m=0.1$.

reach "memory" fixed points starting from initial configurations of complete networks, which are noisy versions of the memories to be learned. A *basin of attraction* is the region of network state configurations around each memory such that if the network begins in the basin, it evolves to the fixed point. The network was trained using examples of the memories at the same level of noise, and a search was made for the energy function $g(x)$, such that training the network by minimizing an energy

$$E(\mathbf{J})=\sum_\mu g(\sqrt{N}\,\boldsymbol{\xi}_o^\mu\mathbf{J}\cdot\boldsymbol{\xi}^\mu) \tag{5.15}$$

would give the largest basins of attraction. For low noise the best g generates the MSA (Sec. III.B.1), but for high noise (i.e., low m), remarkably, the best g is linear, which generates the Hebb algorithm, Eq. (2.17). For intermediate values of noise the optimal $g(x)$ must be found numerically from a Maxwell construction (Huang, 1987).

The same analysis applies to the proximity problem, with prototypes taking the place of memories to be retrieved (Watkin, Rau, Bollé, and van Mourik, 1992). The choice of energy function leads to a *sculpture* of an "energy surface" in input space, where basins of attraction correspond to areas of input space around prototypes which give the correct answer. The maximum stability rule generates narrow, deep valleys in the energy surface around each example presented; so they are well stored. A Hebb rule, by contrast, generates wider, shallower valleys, so that although each example is not stored perfectly (ϵ_t is finite), its influence affects a wider region. If noise is high (the case principally studied by Hansel and Sompolinsky, 1990), the second rule is preferable (Watkin, Rau, Bollé, and van Mourik, 1992), so that a wide valley evolves around each prototype. This is demonstrated by line 4 in Fig. 17, where the prediction for the Hebb algorithm is calculated by a simple geometric method in the manner of Sec. II.D. The points show the results of a single numerical simulation using $N=5000$ and $m=0.1$. The optimal choice of an energy function may be a technique with much application in other optimization problems. In fact, it is possible to show by a simple geometrical argument (Watkin, 1991) that for low m the Hebb algorithm is optimal learning, under the definition in Sec. II.G. In this case version space and student space have almost nothing in common.

The multiclass analog of the proximity problem, in which prototypes are associated with several answers, has been shown (Watkin, Rau, Bollé, and van Mourik, 1992) to be solved more efficiently by a single multiclass perceptron than by the simplest combination of binary perceptrons, which has the same number of possible output states. This is in accordance with the general philosophy of Sec. V.E.

VI. MULTILAYER LEARNING

We have now summarized the theory of perceptron learning. Engineering applications of neural networks

usually require multilayer networks, however, because the real rules they are supposed to learn are not linearly separable. In this section we describe extensions of the statistical theory of learning to multilayer networks (MLN). This is presently the most important direction of new research.

A. Architectures

The simplest generalization of a perceptron has an architecture with one hidden layer. Two such networks are shown in Fig. 18. Each network has K hidden units labeled by the variable $k \in \{1, \ldots, K\}$. Using the nomenclature of Sec. II.A, we can say that the network in Fig. 18(a) is treelike, because no two hidden units receive signals from the same input neurons. The hidden units are said to have *nonoverlapping receptive fields.* The kth hidden unit receives signals from input neurons $i \in [N(k-1)/K]+1, \ldots, Nk/K$. In the network of Fig. 18(a) $N=9$ and $K=3$.

By contrast, the network in Fig. 18(b) is only "feedforward", because its hidden units have overlapping receptive fields. All K hidden units receive inputs from all N input neurons. Such a network is also called *fully connected.* Under many learning strategies the connections to different hidden units will become correlated; so such a network will be more difficult to analyze.

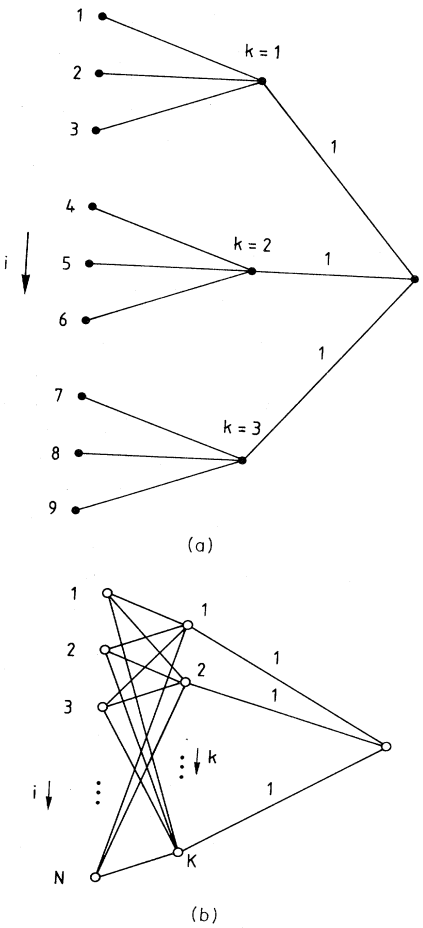

FIG. 18. A committee machine: (a) a treelike network. Each of the K hidden units receives inputs from three input neurons; (b) a fully connected network.

Clearly, the possibility of hidden layers leads to an enormously enlarged student space. In fact, it has been shown that a system with one output mode but arbitrarily many nodes in a single hidden layer can perform any Boolean function of its inputs, if all the neurons have a binary output (De Figueiredo, 1980; Hecht-Nielsen, 1987), and can perform any real function of the inputs if all the nodes have continuous outputs (Denker *et al.*, 1987).

The networks in Fig. 18 have two distinct types of connection: from the input layer to the hidden layer, and from the hidden layer to the output. In many learning applications all of these connections may be adjusted, but the term "machine" has been coined for networks in which the function which the output layer performs of the hidden layer is fixed. Thus the only degrees of freedom in machines are the connections between the input layer and the hidden layer.

One such network is the committee machine. This has K odd, and the output neuron is set to agree with the majority of the hidden units,

$$S_o = \mathrm{sgn}\left[\sum_{k=1}^{K} S_k\right] . \tag{6.1}$$

Thus, for a committee machine with nonoverlapping receptive fields, Fig. 18(a),

$$S_o = \mathrm{sgn}\left[\sum_{k=1}^{K} \mathrm{sgn}\left[\sum_{i=[N(k-1)/K]+1}^{Nk/K} J_{k,i} S_i\right]\right] , \tag{6.2}$$

where, following Sec. II.A, $J_{k,i}$ is the strength of the weight from the ith input to the kth hidden unit.

For a machine with overlapping receptive fields, Fig. 18(b),

$$S_o = \mathrm{sgn}\left[\sum_{k=1}^{K} \mathrm{sgn}\left[\sum_{i=1}^{N} J_{k,i} S_i\right]\right] . \tag{6.3}$$

Function Eq. (6.1) is of the form Eq. (2.2), so the output neuron performs "synaptic emulation" with all the weights between the hidden layer and the output set equal to $+1$.

Any function which can be performed by a two-layer network of binary neurons with Ising weights and without thresholds can also be performed by a committee machine with Ising weights, since, if in any network with these kind of states and weights and the architecture of Fig. 18, the weight $J_{o,k}$ between a hidden-layer node k and an output node o were -1, then the state of the output node would be invariant under the transformation $J_{o,k} \mapsto 1$ and $J_{k,i} \mapsto -J_{k,i}$ for all i. Thus the committee machine is the most general two-layer machine of Ising weights in which every neuron performs synaptic emulation.

A slightly different machine is the parity machine

(Mézard and Patarnello, 1989; Mitchison and Durbin, 1989). In this case the output is set equal to the parity of the hidden units

$$S_o = \left[\prod_{k=1}^{K} S_k \right] . \tag{6.4}$$

Note that in this case the output neuron is not performing synaptic emulation, Eq. (2.2).

Another possible machine is the "AND machine" (Zollner *et al.*, 1992), in which the output neuron is only $+1$ is all the hidden units are $+1$.

We might wonder how much our network will be restricted if we make it a "machine" by fixing the weights between the hidden layer and the output. If $K \ll N$, then one might expect that this restriction will be small, because only K degrees of freedom are removed, leaving of order N. The problem seems to be a difficult one, however, and is still under investigation (Grossman and Domany, 1992, private communication).

Does a simple analog exist for multilayer networks of the (**B-J**) space diagram, used in Sec. II.D? We can see from Eqs. (6.2) and (6.3) that each of the hidden units in a committee machine divides the input space by a hyperplane. The output neuron then selects certain regions of the resulting partition and gives these regions output $+1$. We can represent this schematically as in Fig. 19, which shows the input space of a network with three hidden units. Each of the three hyperplanes marks the division of the input space into the region where one hidden unit is $S_k = +1$ and the region where that hidden unit is $S_k = -1$. In this diagram the output of the committee machine is marked (as ± 1) in each region of the input space. Clearly this is a more complex partition than that made by a perceptron.

Unfortunately, unlike the (**B-J**) diagram of Sec. II.D, it is difficult to draw this diagram so as to extract *quantitative* information. In reality it should be drawn in more than two dimensions, because the vectors of connection strengths to the hidden units span a K-dimensional space, and important directions to be learned will add further dimensions.

For multilayer networks there is a new fundamental concept. Although a given example of a rule sets the states of the input and output nodes, it does not itself place restrictions on the corresponding states of nodes in the hidden layers. The state of these "internal" nodes in a network $\mathcal{N}$ for each possible state **S** of the input nodes is $\mathcal{N}$'s *representation* of **S**. Clearly any two inputs which are supposed to be mapped to different outputs must be represented by different configurations of states on *every* hidden layer. Such a representation is called *faithful* (Mézard and Nadal, 1989).

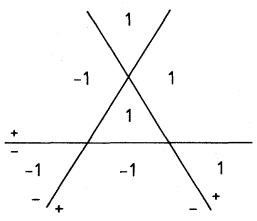

FIG. 19. Schematic diagram of how the committee machine with $K=3$ partitions input space. Each hidden unit has a positive state for a question on the $+$ side of one hyperplane. In each region of input space, the output of the machine is marked as 1 or -1.

B. Storing memories in multilayer networks

Elsewhere in this article we have concentrated on the problem of learning a rule with a network. Historically, however, the first neural network problem to which statistical mechanics was applied was that of "storing memories." By this we mean designing a network which will memorize a set of input configurations, so that whenever one of these configurations is applied, the network will give a prearranged answer.

Presently there is little understanding of how a multilayer network learns a rule, which we believe to be the most important open question in the field. This section of the paper therefore seeks to develop some intuition by describing recent work on storing memories in multilayer networks.

To emphasize the difference between the concepts in this section and the concepts in the rest of the review, we shall label all quantities which are analogous to those in learning theory by a superscript line. For example, we shall be considering how a network can memorize $\bar{p}$ memories.

Formally, storing memories means finding a network $\mathcal{N}$, such that when any of the $\bar{p}$ configurations $\{\boldsymbol{\chi}^\mu\}$, $\mu = 1, \ldots, \bar{p}$, is presented as input, the network will give the respective output χ_o^μ. Thus $\mathcal{N}(\boldsymbol{\chi}^\mu) = \chi_o^\mu$ for all μ.

This problem is quite different from that of learning a rule: we do not need to find the answers to *new* questions. When we are just storing memories there is no underlying rule to be found, and in fact it is usual to assume that the inputs $\boldsymbol{\chi}^\mu$ and outputs χ_o^μ are uncorrelated.

First, we make a short excursion from the main topic of the article to review how geometrical arguments can be used to place rigorous bounds on the number of memories which may be stored in a network of a given architecture (Winder, 1963; Cover, 1965; Venkatesh, 1986; Mitchison and Durbin, 1989). This is included because we know of no other easy explanation of the results. However, it should perhaps be omitted on a first reading of the paper.

In Sec. VI.B.2 we discuss how statistical mechanics solves the same problem.

1. The geometrical approach—an excursion

We define a dichotomy as any division of the set of memories into two groups by a hyperplane through the origin. For example, in Fig. 20(a), four questions $\boldsymbol{\chi}^1$–$\boldsymbol{\chi}^4$ are marked in a two-dimensional space. A hyperplane in

any space is $(N-1)$ dimensional, so in this case it is a straight line. The hyperplane also has a positive and a negative side. One dichotomy is marked in Fig. 20(a), dividing the memories so that pattern 1 is on the positive side of the plane, while the rest are on the other side. We call the resulting partition $\{\underline{\xi}^1\};\{\underline{\xi}^2,\underline{\xi}^3,\underline{\xi}^4\}$. We count $\{\underline{\xi}^2,\underline{\xi}^3,\underline{\xi}^4\};\{\underline{\xi}^1\}$ as a different dichotomy. Having all the points on the same side of the hyperplane is a dichotomy (if it can be achieved), and all the points on the other side is another dichotomy. It is easy to see that there are eight possible dichotomies of the four memories in this two-dimensional space.

In general, we call the number of dichotomies of $\bar{p}$ memories in N dimensions $C(\bar{p},N)$. Clearly, in the example above, $C(4,2)=8$, and this does not depend upon where the memories lie in the two-dimensional space (provided that no two memories lie in the same direction from the origin). In general, $C(\bar{p},N)$ is the same for *any* set of memories, provided that no l memories (with $l \le N$) are linearly dependent.

Of course, if we want to memorize a response for each of these memories, we must find a dichotomy such that all the memories with the answer $+1$ are on one side of the hyperplane, and all the memories with answer -1 are on the other side. Since the total number of possible divisions of $\bar{p}$ patterns into two sets is $2^{\bar{p}}$, the chance that one of the $C(\bar{p},N)$ dichotomies we can produce is the one we want is $f \equiv C(\bar{p},N)/2^{\bar{p}}$. We shall now show that $f=\frac{1}{2}$ for $\bar{p}=2N$.

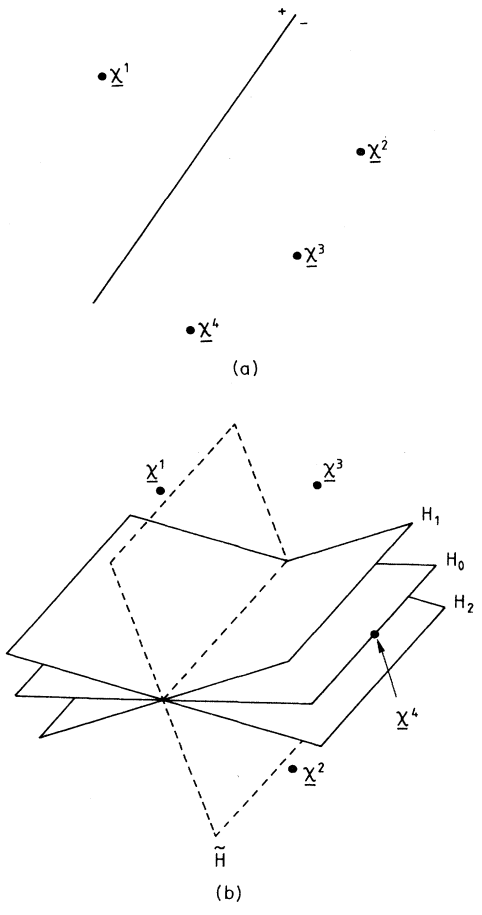

FIG. 20. Counting dichotomies: (a) A dichotomy of four points. (b) Each of hyperplanes H_1 and H_2 produce the same dichotomy of the first $\bar{p}$ planes, but different dichotomies when the $(p+1)$st pattern is added. Hyperplane H_0 is parallel to $\xi^{\bar{p}+1}$. $\tilde{H}$ is perpendicular to $\xi^{\bar{p}+1}$.

We do this by induction. We have $C(1,N)=2$, because in any number of dimensions one point can lie on either of two sides of a hyperplane. Similarly, $C(\bar{p},1)=2$ because in one dimension the only possible hyperplane is the origin itself, and a set of memories along the x axis can be regarded as having two possible orientations about this point.

Now we consider all the dichotomies which may be formed of the first $\bar{p}$ memories in N dimensions and calculate how many *more* may be formed when the $(\bar{p}+1)$th memory is added.

Figure 20(b) shows a schematic diagram of two hyperplanes H_1 and H_2, which produce the same dichotomy D of the first $\bar{p}$ patterns (the figure shows $\bar{p}=3$). Pattern $\xi^{\bar{p}+1}$, however, lies on different sides of the two planes, so that the planes produce a different dichotomy of the set of $\bar{p}+1$ memories. It is clear that there exists a hyperplane H_0 which also produces a dichotomy of the first $\bar{p}$ memories and which is parallel to $\xi^{\bar{p}+1}$. It is easy to see that $C(\bar{p}+1,N)=C(\bar{p},N)+\Gamma$, where Γ is the number of dichotomies which can be formed of the first $\bar{p}$ patterns by hyperplanes parallel to $\xi^{\bar{p}+1}$.

Now consider projecting this diagram onto the hyperplane $\tilde{H}$ perpendicular to $\xi^{\bar{p}+1}$. The projection of H_0 forms a dichotomy of the projection of the first $\bar{p}$ memories in this $(N-1)$-dimensional space. Therefore it is easy to see that Γ is just $C(\bar{p},N-1)$.

Thus we have

$$C(\bar{p}+1,N)=C(\bar{p},N)+C(\bar{p},N-1)\ . \tag{6.5}$$

This, and the values $C(1,N)$ and $C(\bar{p},1)$ given above, is enough to prove that

$$C(\bar{p},N)=2\sum_{j=0}^{N-1}\binom{\bar{p}-1}{j}\ . \tag{6.6}$$

It follows that $f \equiv C(\bar{p},N)/2^{\bar{p}}$ is the sum of the first N terms of the $\bar{p}$ terms in the binomial expansion of $(\frac{1}{2}+\frac{1}{2})^{\bar{p}-1}$. Thus $f=\frac{1}{2}$ when $\bar{p}=2N$.

In fact, as N becomes large, f effectively changes from being 1 to 0 when $\bar{\alpha} \equiv \bar{p}/N$ moves through a range of values of width $\sim 1/\sqrt{N}$ about $\bar{\alpha}=2$. Therefore a large spherical perceptron can memorize two memories per synapse (Winder, 1963; Cover, 1965; Venkatesh, 1986).

Note, in passing, that Eq. (6.6) can be shown to be $2^{\bar{p}}$ for $\bar{p}<N$, and less than $2^{\bar{p}}$ for higher $\bar{p}$. This proves the fact, stated in Sec. II.E, that the VC dimension of a perceptron (defined in Sec. II.E) is N.

How far is it possible to generalize this argument to multilayer networks? Using Eqs. (6.2) and (6.3) we can

consider each of the hidden units as making a linear partition of the input space. Is it possible to generalize the concept of a dichotomy?

The difficulty is illustrated in Fig. 21. If we consider the six memories shown in a two-dimensional space in Fig. 21(a), then *any* three points a, b, and c can be partitioned from the others by two hyperplanes. However, if the six points are arranged in a hexagon, then there are some sets of three points which cannot be divided from the rest by two hyperplanes, as illustrated in Fig. 21(b). We conclude that for spaces divided by many planes, there is no obvious analog of $C(\bar{p},N)$ which does not depend upon the positions of the memories; for multilayer networks, we can only measure the *average* number of patterns which can be stored when patterns are drawn from some distribution. This is the thermodynamic approach considered in the next section.

However, it is possible to use simple arguments to place *bounds* on the number of examples which can be stored (Mitchison and Durbin, 1989). Suppose we are trying to memorize patterns in the network of Fig. 18(a) in which the hidden units have nonoverlapping receptive fields. We can define $I(\{\boldsymbol{\chi}^\mu,\chi_o^\mu\},K,N)$ as the number of ways in which a multilayer network can divide a certain set of $\bar{p}$ patterns into two sets. Then, using the argument developed for the perceptron, the chance that the network can learn a given output for this set of patterns is $f \equiv I(\{\boldsymbol{\chi}^\mu,\chi_o^\mu\},K,N)/2^{\bar{p}}$, which is one-half for a critical number of patterns $\bar{p}_c$, which we call the capacity. We define $\bar{\alpha}_c$ as $\bar{p}_c/N$.

Each hidden unit has N/K inputs so it can implement $C(\bar{p},N/K)$ dichotomies of $\bar{p}$ patterns. Thus an upper bound on the number of ways in which the K planes can partition $\bar{p}$ patterns is $C(\bar{p},N/K)^K$. Therefore $I(\{\boldsymbol{\chi}^\mu,\chi_o^\mu\},K,N) \le C(\bar{p},N/K)^K$.

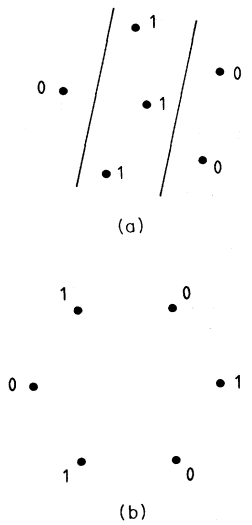

FIG. 21. Partitioning six points: (a) Any set of points in this arrangement can be partitioned from the others by two hyperplanes (lines). An example is given. (b) No two hyperplanes (lines) can partition the six points, so that those marked 0 are separated from those marked 1.

Barkai *et al.* (1992) have pointed out that $C(\bar{p},N/K)$ can be expressed as an integral, which can then be evaluated for large N by the saddle-point method, to give

$$C(\bar{p},N/K) \propto (\bar{\alpha}K)^{\bar{p}}(\bar{\alpha}K-1)^{(\bar{p}-N/K)} . \tag{6.7}$$

Substituting this into the condition for $\bar{p}_c$ above gives an upper bound on $\bar{\alpha}_c$ as the solution of the equation

$$\bar{\alpha}_c K = (\bar{\alpha}_c - 1)^{[1-(1/\bar{\alpha}_c K)]} 2^{1/K} . \tag{6.8}$$

This gives $\bar{\alpha}_c \le 5.43$ for $K=3$. As $K \to \infty$, we obtain $\bar{\alpha}_c \le \mathcal{O}(\ln K)$.

We can generalize this argument to the case of the network in Fig. 18(b) with overlapping receptive fields. In this case the hidden units perform dichotomies of the $\bar{p}$ patterns in N dimensions; but otherwise the arguments are the same, and the upper bound for $I(\{\boldsymbol{\chi}^\mu,\chi_o^\mu\},N,K)$ becomes $(C(\bar{p},N))^K$. This leads to upper bounds for the critical capacity which are just K times as high as the ones for overlapping receptive fields. That is, $\bar{\alpha}_c \le 16.29$ for $K=3$, and as $K \to \infty$, $\bar{\alpha}_c \le \mathcal{O}(K \ln K)$ (Mitchison and Durbin, 1989; Barkai *et al.* 1992).

2. A statistical-mechanics approach

To calculate exactly the storage capacity of a multilayer network, it is necessary to turn from geometrical arguments to algebraic ones.

We noted in Sec. III.A.2 that in the zero-temperature limit ($\beta \to \infty$) the partition function Eq. (3.3) becomes equal to the fraction of the student space which correctly stores all examples of a rule. Similarly, in the problem of storing memories it is natural to consider the part of the student space which correctly remembers all the memories. Let us call it the "parrot space" $\mathcal{P}$. In the case of the perceptron, $\mathcal{P}$ is a fraction

$$\bar{Z} = \int d\mu(\mathbf{J}) \prod_\mu \Theta(\sqrt{N}\,\chi_o^\mu \mathbf{J}\cdot\boldsymbol{\chi}^\mu) \tag{6.9}$$

of the student space.

This is a volume in an N-dimensional space, so it is exponentially large. Different choices of examples make $\bar{Z}$ different by a large factor, and so that average of $\bar{Z}$ over the sets of memories which could be stored, which we denote by $\langle \bar{Z} \rangle_\chi$, will be dominated by exponentially unlikely sets of examples which make $\bar{Z}$ largest. It is necessary to study instead $\langle \ln \bar{Z} \rangle_\chi$.

The calculation of $\langle \ln \bar{Z} \rangle_\chi$ may be performed in the same way as that presented in Sec. III.A (Gardner, 1988). The only difference is that outputs $\{\chi_o^\mu\}$ are uncorrelated with the inputs $\{\boldsymbol{\chi}^\mu\}$, instead of being related through a rule. In this case a parameter $q^{\gamma'\gamma}$ measures the correlation, $\mathbf{J}^\gamma \cdot \mathbf{J}^{\gamma'}$, of two perceptron vectors $\mathbf{J}^\gamma$ and $\mathbf{J}^{\gamma'}$ chosen randomly in $\mathcal{P}$. If the student space consists of all spherical $\mathbf{J}$ vectors, the parrot space is connected and convex, since it is the part of the continuous space obeying $\bar{p}$ linear constraints and the normalization condition. Thus we expect that replica symmetry is obeyed: $q^{\gamma'\gamma}$ is sharp-

ly peaked at a single value $q(\bar{\alpha})$; this result was derived by Gardner and Derrida (1988).

When no memories are stored ($\bar{\alpha}=0$), the parrot space is all normalized **J** vectors; thus $q(\bar{\alpha}=0)=0$. As α rises, the parrot space shrinks, and as $\bar{\alpha}\to 2$, $q(\bar{\alpha})\to 1$: the parrot space shrinks to a point. Thus the maximum number of memories which may be learned by a perceptron is 2, in agreement with the geometrical argument of Sec. VI.B.1.

a. Memory storage in machines with nonoverlapping receptive fields

This calculation has been extended to storing memories in a committee machine (Barkai *et al.*, 1992; Engel *et al.*, 1992) and in a parity machine (Barkai *et al.*, 1990). The analog of (6.9) for a committee machine, Eq. (6.2), with nonoverlapping receptive fields is

$$\bar{Z}=\int d\mu(\mathbf{J})\prod_{\mu=1}^{p}\Theta\left\{\chi_o^\mu\sum_{k=1}^{K}\operatorname{sgn}\left[\sqrt{K/N}\sum_{i=[N(k-1)/K]+1}^{Nk/K}J_{k,i}\chi_i^\mu\right]\right\}, \tag{6.10}$$

where the measure on the weight space is

$$d\mu(\mathbf{J})\equiv\prod_{k=1}^{K}\prod_{i=[N(k-1)/K]+1}^{Nk/K}dJ_{k,i}\prod_{k=1}^{K}\delta\left[\sum_{i=[N(k-1)/K]+1}^{Nk/K}J_{k,i}^2-\frac{N}{K}\right]. \tag{6.11}$$

Equation (6.10) reduces to (6.9) in the case of $K=1$, because a committee machine with one hidden unit is just a perceptron. The analog of (6.10) for a parity machine differs only in that the sum over hidden units inside the Θ function is replaced by a product over hidden units.

As before, we shall calculate $\langle \ln\bar{Z}\rangle_\chi$; hence we replicate the system and consider $\langle \bar{Z}^n\rangle_\chi$. After averaging over the memories, in the manner of Sec. III.A.7, the expression for $\langle \bar{Z}^n\rangle_\chi$ contains the order parameter

$$q_k^{\gamma'\gamma}\equiv\left\langle\left\langle\frac{K}{N}\sum_{i=[N(k-1)/K]+1}^{Nk/K}J_{k,i}^{\gamma}J_{k,i}^{\gamma'}\right\rangle_\beta\right\rangle_\chi,\quad \gamma\neq\gamma', \tag{6.12}$$

where $\langle\cdots\rangle_\beta$ means the average over the parrot space. $q_k^{\gamma'\gamma}$ is the average overlap of the connections to the kth committee member in two different replicas. Two symmetries may be present: replica symmetry ($q_k^{\gamma'\gamma}=q_k$ $\forall\gamma\neq\gamma'$) and committee symmetry ($q_k^{\gamma'\gamma}=q^{\gamma'\gamma}$ $\forall k$). Under both assumptions $\bar{Z}^n$ may be written as a function of $\bar{\alpha}$ and of just one order parameter q, which is found from the saddle-point equations.

Both for the committee machine and for the parity machine the function $q(\bar{\alpha})$ rises from 0, at $\bar{\alpha}=0$, to $q=1$ at a critical value $\bar{\alpha}_c$, which is the capacity of the network. Barkai *et al.* (1992) and Engel *et al.* (1992) show that for the committee machine with $K=3$, $\bar{\alpha}_c\approx 4.02$, while Barkai *et al.* (1990) show that for the parity machine and $K=3$, $\bar{\alpha}_c\approx 10.3$. The second of these results violates the rigorous upper bound obtained in Sec. VI.B.1. For both machines RS gives $\bar{\alpha}_c\sim K^{1/2}$ as $K\to\infty$, which also violates the upper bound.

How can we understand this? Suppose that we have chosen a representation R of the patterns; that is, for each χ^μ, we have chosen what states all the K units in the hidden layer will take. Since R defines a set of $\bar{p}$ input-output relations which each of the K hidden units must memorize, the vector of connection strengths to a given hidden unit must be part of a connected region. Thus the space of networks which give representation R is the product of these connected regions, a *bubble* in student space.

Different bubbles correspond to different representations. Note that the bubbles may be of different sizes, and indeed it may be true that some internal representations try to make a single committee member store a set of input-output relations which cannot be memorized. For such a representation, the volume of the bubble will be zero.

Since the student space is disconnected, replica symmetry is strongly broken. Two randomly chosen networks may be in the same bubble (in which case they will be highly correlated), or in different bubbles. Barkai *et al.* (1992) and Engel *et al.* (1992) analyzed first-step replica symmetry breaking in the committee machine and found that replica symmetry is stable only for $\bar{\alpha}<1.76$, while the corrected critical capacity is $\bar{\alpha}_c^{\mathrm{RSB}}\approx 3.0$. In the large K limit, the capacity $\bar{\alpha}_c^{\mathrm{RSB}}$ scales as $\ln K$, which agrees with the bound of Sec. VI.B.1.

For the parity machine, $\bar{\alpha}_c^{\mathrm{RSB}}\approx 5.0$ for $K=3$ (Barkai *et al.*, 1990). For the case of $K\gg 1$, first-step replica symmetry breaking may give the exact capacity for the parity machine, since $\bar{Z}$ may be written in a form resembling the partition function of a spin glass with multispin interactions, for which first-step RSB is known to be exact (Derrida, 1981; Gross and Mézard, 1984).

Kanter (private communication, 1992) has also given a simple explanation of why the storage capacity of a parity machine is greater than that of a committee machine. Storing a memory in a committee machine requires a representation in which the majority of the hidden units have state χ_o^μ. By contrast, the representations in a parity machine need only have the product of the states of the hidden units equal to χ_o^μ, which is a much weaker constraint.

Note also that in Fig. 19 large fractions of the planes

are redundant, in the sense that the output of the machine is the same for questions on different sides of the plane. By contrast, in the parity machine, every plane always divides regions of input space which give different outputs. The parity machine therefore makes a more efficient partition of input space (Mitchison and Durbin, 1989).

b. Memory storage in machines with overlapping receptive fields

The storage of memories in a committee machine with overlapping receptive fields is considerably more complex than in the nonoverlapping case. Since all hidden units receive signals from the same input nodes, they become correlated during the process of learning.

Note that the parrot space possesses an interesting *permutation symmetry*. The output of the committee machine is invariant with respect to permutations of the hidden units. Thus if Σ is a $K\times K$ permutation matrix and if $\{J_{k,i}\}$ is a point in the parrot space, then so is $\{J'_{k,i}\}$, where $J'_{k,i}=J_{\Sigma(k),i}$ for all k.

Some progress has been made by Barkai *et al.* (1992) and Engel *et al.* (1992) in analyzing this model. The volume of the parrot space is

$$\bar{Z}=\int d\bar{\mu}(\mathbf{J})\prod_{\mu=1}^{p}\Theta\left\{\chi_o^\mu\sum_{k=1}^{K}\operatorname{sgn}\left[\frac{1}{\sqrt{N}}\sum_{i=1}^{N}J_{k,i}\chi_i^\mu\right]\right\}, \tag{6.13}$$

where the measure on the weight space is

$$d\bar{\mu}(\mathbf{J})\equiv\prod_{k=1}^{K}\prod_{i=1}^{N}dJ_{k,i}\prod_{k=1}^{K}\delta\left[\sum_{i=1}^{N}J_{k,i}^2-N\right]. \tag{6.14}$$

An analysis of $\langle\bar{Z}^n\rangle_\chi$ gives a result in terms of three significant order parameters

$$q_k^{\gamma'\gamma}\equiv\langle\langle J_k^\gamma\cdot J_k^{\gamma'}\rangle_\beta\rangle_\chi,$$

$$C_{k,k'}=\langle\langle J_k^\gamma\cdot J_{k'}^\gamma\rangle_\beta\rangle_\chi,$$

and

$$D_{k,k'}^{\gamma\gamma'}=\langle\langle J_k^\gamma\cdot J_{k'}^{\gamma'}\rangle_\beta\rangle_\chi.$$

If we assume that all these parameters are replica symmetric (that is, the replica indices can be neglected), they can be interpreted, using Sec. III.A.7, as meaning,

$$q_k\equiv\langle\langle J_k\rangle_\beta^2\rangle_\chi,$$

$$C_{k,k'}=\langle\langle J_k\cdot J_{k'}\rangle_\beta\rangle_\chi,$$

and

$$D_{k,k'}=\langle\langle J_k\rangle_\beta\cdot\langle J_{k'}\rangle_\beta\rangle_\chi.$$

Barkai *et al.* (1992) have shown that for small $\bar{\alpha}$, solutions which are related by permutation symmetry are in the same connected space of solutions, and thus in the same ergodic component. Therefore for every realization of the patterns, $\langle J_k\rangle_\beta$ is the same for all hidden units and $q=D$. This is called the permutation-symmetric (PS) phase.

As $\bar{\alpha}$ rises, however, the parrot space shrinks, and there is a second-order transition in which permutation symmetry is broken and q and D diverge. Both the PS phase and the PS-breaking phase have been observed by measuring the temporal correlations of a zero-temperature Monte Carlo simulation inside the parrot space.

Engel *et al.* (1992) have shown that as $\bar{\alpha}$ rises further to a critical value of $\bar{\alpha}_c\approx 34.5$, q rises to 1. Thus this $\bar{\alpha}_c$ is taken to be the replica-symmetric capacity of the network. Unfortunately, the capacity is more than twice the rigorous upper bound obtained in Sec. VI.B.1; so replica symmetry, as well as permutation symmetry, must be strongly broken in this model. The numerics for first-order replica symmetry breaking seem to be very difficult, so that so far the best estimates we have for the storage capacity are obtained by simulation, using algorithms described in the next section. For $K=3$, the best known algorithm is able to achieve a storage capacity of $\bar{\alpha}_c\approx 3\times(2.82\pm 0.02)$ (Barkai *et al.*, 1992).

Although replica symmetry is broken in this model, we expect many qualitative features of the solution to be valid for the RSB result. In particular, the RS solution shows that as $\bar{\alpha}\to\bar{\alpha}_c$, D and C both tend to $1/(1-K)$, which implies that the committee members become negatively correlated. This result can easily be understood (Engel *et al.*, 1992): at $\bar{\alpha}_c$ most patterns have internal representations with $(K+1)/2$ of the hidden units equal to χ_o^μ, and $(K-1)/2$ opposite. If we select two hidden units from the K, the chance that they have different outputs is $2\times[(K+1)/2K]\times[(K-1)/2(K-1)]=(K+1)/2K$, which is greater than $\frac{1}{2}$. This gives rise to an anticorrelation of two different committee members of order of $1/K$. $D<0$ demonstrates that there is a division of labor among the hidden units: each hidden unit attempts to learn the patterns which have not been learned by the others.

As $K\to\infty$, C and D tend to zero, so that the correlations between hidden units become negligible. $\bar{\alpha}_c$ for the fully connected network is then just K times $\bar{\alpha}_c$ for the treelike one. This means that, in this limit, the capacity per synapse given a network with overlapping receptive fields becomes equal to that of one with nonoverlapping receptive fields (Engel *et al.*, 1992).

3. Backpropagation and other memorizing algorithms

The stochastic formulation of Sec. III.A.2 applies, as was pointed out, to any fixed network architecture which can be completely described by a vector $\mathbf{J}$. However, stochastic algorithms are very slow, and in practice it is always preferable to minimize energies by a gradient descent search of student space. In multilayer networks the derivatives of a differentiable energy $E(\mathbf{J})$ with respect to

the weights in a certain layer are, by the chain rule, functions of the weights in later layers (closer to the output). Therefore if the energy used is the training energy, Eq. (2.11), errors are said to be "propagated back" when the network is trained; hence the algorithm is known as *backpropagation* (BP). This is by far the most common approach to training multilayer networks (Bryson and Ho, 1969; Le Cun, 1986; Rumelhart and McClelland, 1986; Lippman, 1989). It has been used in numerous applications, such as data compression or encoding (Ackley *et al.*, 1985; Cottrell *et al.*, 1987), pronunciation of written text (Sejnowski and Rosenberg, 1987), speech recognition (Waibel *et al.*, 1989), and robotics (Eckmiller, 1989).

To increase the speed of learning, more sophisticated minimization procedures can be applied, for example, conjugate gradient methods (Kramer and Sangiovanni-Vincentelli, 1989; Mackram-Ebeid *et al.*, 1989) or quasi-Newton algorithms (Watrous, 1987). A problem common to all these approaches is the many local minima in an energy landscape such as that defined by Eq. (2.11). Any descent method can get trapped in a local minimum, which may occur at a rather high value of $E_t(\mathbf{J})$. When such trapping occurs can often be predicted using statistical mechanics, by considering when the stochastic algorithms of Sec. III.A.2 experience replica symmetry breaking. Convergent learning algorithms are usually known in the region of phase space characterized by replica symmetry, while no learning algorithm is known to converge in a bounded learning time in the region with replica symmetry breaking.

The absence of wide flat regions in the energy surface makes learning significantly faster. Such well-formed energy functions (Wittner and Denker, 1988) have been studied by, for example, Solla *et al.* (1988) and Fahlmann (1989).

In an attempt to overcome this difficulty, one may introduce noise to the learning process (e.g., Sietsma and Dow, 1988; von Lehmann *et al.*, 1988); and thus allow increases of $E(\mathbf{J})$ in order to overcome the energy barriers of a local minimum. Alternatively, a *momentum* may be introduced so that $\mathbf{J}$ has an inertia and a high-energy barrier is required to deflect its motion (Plaut *et al.*, 1986; Jacobs, 1988; Vogl *et al.*, 1988).

No example of such learning has been solved using statistical mechanics. However, Eisenstein and Kanter (1992) recently presented an interesting partly analytic and partly numerical study of backpropagation for a tree-like parity machine. They used an energy E that is a smoothed form of the training energy, but also such that $E(\mathcal{N})=0$ only when all memories are correctly stored. For $\bar{\alpha}<1/(2K)$ they were able to show that no spurious local minima of the energy function exist, and for higher α simulations show that either no local minima exist or their number and size is small.

The capacity of the parity machine with $K=3$ calculated by thermodynamics using an energy E_t is $\bar{\alpha}=3.9$, with two steps of RSB. Numerically, it seems that this capacity is saturated. This raises two possibilities: either backpropagation is able to converge in an RSB region (suggesting that even if there is RSB, there are no local minima), or else using a smooth energy function has removed RSB without altering the capacity of the network.

The convergence time $t_{\rm con}$ of the BP algorithm diverges as $\alpha\to\alpha_c$ as $t_{\rm con}=a(N)(\alpha_c-\alpha)^{-2}$, with $a(N)$ being almost independent of N, rendering a convergence time which is not exponential with N.

We should also mention that besides backpropagation a number of other algorithms exist for storing memories in multilayer networks, even though none approaches the theoretical storage capacity. In addition, these algorithms cannot be framed as a stochastic process.

One algorithm especially appropriate for the parity machine is the least action algorithm (LAA; Mitchison and Durbin, 1989), in which we try to make the network correctly memorize one pattern at a time by altering the connections to one hidden unit (the hidden unit whose connections have to be altered least to memorize the pattern). By repeating this process we try to minimize the number of misremembered patterns.

For networks of binary neurons, several schemes have been proposed (Grossman *et al.*, 1989; Grossman, 1990; Krogh *et al.*, 1990; Nabutovsky *et al.*, 1990; Rohwer, 1990; Saad and Marom, 1990), which seek optimal representations, rather than focusing on the connections themselves. Of course, once a representation is chosen, the nodes have been decoupled, in the sense that since the desired reaction of each node to its inputs is now specified, the functions nodes perform can be trained independently. For example, the CHIR algorithm (choice of internal representation), proposed by Grossman *et al.* (1989), selects a certain internal representation of the training set and tries to adjust the weights of inputs to each node using the simple perceptron algorithms (Sec. III). If this attempt fails, another internal representation is selected, so that eventually the number of wrongly mapped patterns is minimized.

C. Learning a rule in multilayer networks

This article is concerned with the ability of neural networks to generalize. Since most learning problems are not linearly separable, it is of great importance to understand how multilayer networks perform this task.

In this subsection we describe how MLN learn certain rules. We start by considering MLN used to learn linearly separable rules and proceed to more complex rules which can only be implemented on networks with at least one hidden layer.

1. MLN learning a linearly separable rule

Recently two examples of MLN learning linearly separable rules have attracted attention. Opper and Haussler (1991a) have shown that the Bayesian optimal algorithm can be implemented exactly by a large committee machine. Schwarze, Opper, and Kinzel (1992) ana-

lyzed a committee machine learning a linearly separable rule, corroborating a notion analogous to Occam's razor that the best network to use is the simplest one capable of learning a specific rule.

a. Implementing the Bayes algorithm

As noted in Sec. III.C.1, the Bayes optimal generalization can be implemented exactly by a single perceptron student learning a linearly separable rule in the limit of high N. Opper and Haussler (1991a) have pointed out that even for N small, Bayes algorithm can be implemented by a committee machine in which each node of the hidden layer has been treated as a perceptron and its connections from the input layer trained independently by the method of Sec. III.A, so that each committee member on its own is inside the version space of the rule. In the limit of infinitely many committee members, the entire version space is explored and the output of the committee machine equals the Bayesian prediction.

The committee members in this case are analogous in function to the samplers used in Sec. III.B.2 to train the optimal perceptron, since they sample the version space (although in this case they are part of the network being built, not artifacts of a possible learning algorithm). As explained in Sec. III.C.1, the Bayes algorithm and the optimally trained perceptron differ on a vanishingly small proportion of random questions as $N \to \infty$.

b. The oversophisticated student

The converse problem to learning unlearnable rules, Sec. V.A, is the case in which the student network is more complex than the teacher, so that many points in student space reproduce the rule exactly. Recently Schwarze, Opper, and Kinzel (1992) analyzed the high-temperature generalization behavior of a fully connected committee machine trained from examples of a linearly separable rule. Since a perceptron is capable of learning this kind of rule, this is another example of a mismatched architecture (Sec. V.A.4).

It turns out that the natural order parameters of the system are $P_{kk'} \equiv \mathbf{J}_k \cdot \mathbf{J}_{k'}$, the overlap between two members of the committee (where $\mathbf{J}_k$ means the N-vector of weights of inputs to the kth committee member), and $R_k \equiv \mathbf{J}_k \cdot \mathbf{B}$, the overlap between members of the committee and the rule to be learned. $\epsilon_f(\{P_{kk'}\},\{R_k\})$ and $s(\{P_{kk'}\},\{R_k\})$ may be straightforwardly calculated, whether the $\mathbf{J}_k$ are spherical or Ising (Ising weights to the committee members might be used if the components of $\mathbf{B}$ were Ising). High-temperature learning, explained in Sec. III.A.5, may therefore be applied, simply minimizing Eq. (3.20).

With the simplifying ansatz of committee symmetry ($R_k = R$ for all k, and $P_{kk'} = P$ for all $k \neq k'$), Schwarze (1991) found that the generalization error is always *higher* for a larger number of hidden units. This is an implication of the point made in Sec. II.C, that it is the restrictions on a network which lead to learning. In this case, forcing the committee machine to reproduce the given examples still allows it too much freedom to disagree with the rule on other questions. This suggests the principle that *the best network to use is the simplest capable of learning a rule,* where the discussion of Sec. II.C suggests that the definition of "simplest" should be in terms of minimizing the excess of student space over rule space (rather than, for example, making a tradeoff between minimizing the number of nodes and the total number of connections).

For the case of Ising $\mathbf{J}_k$ and $K = 3$, the possibility was studied of a breaking of committee symmetry, where one of the committee members, say $k = 1$, becomes perfectly aligned with the rule. Thus the order parameters become $R_1 = 1$, $P_{12} = P_{13} = R_2 = R_3 = R$, and $P_{23} = P$. For $\tilde{\alpha} > \tilde{\alpha}_1 \approx 3.15$, this state is the global minimum of the free energy, though the symmetric state persists as a local minimum until $\tilde{\alpha} = \tilde{\alpha}_3 \approx 8.60$. This behavior is shown in Fig. 22(a), where dotted line 1 is the generalization error of the state without the breaking of committee symmetry and dotted line 2 is the state with one committee member aligned. At a value of $\tilde{\alpha} = \tilde{\alpha}_2 \approx 4.77$ the global minimum of the free energy becomes a solution with two of the committee members equal to the teacher: $R_1 = R_2 = P_{12} = 1$ and $P_{13} = P_{23} = R$, the only remaining order parameter; but the solution with only one committee member aligned to the teacher seems to persist until, at least, a very high $\tilde{\alpha}$. Therefore in the limit of an infinite system it will take infinite time to escape from this solution. Since two agreeing members of a three-member committee are enough to determine the committee machine's output, the state with two committee members aligned to the teacher has perfect generalization, $\epsilon_g = 0$, and however large $\tilde{\alpha}$ becomes, the third member will never learn the rule.

Figure 22(a) shows these theoretical predictions (marked as dotted lines) against the results of Monte Carlo simulations performed as explained in Sec. III.A by a single spin-flip dynamics, using $N = 75$ weights at $T = 5$ and with a labeled number of Monte Carlo steps (MCS) per spin. The open symbols (with error bars) show the results with $\tilde{\alpha}$ increasing, and the solid ones with $\tilde{\alpha}$ reducing. In this simulation the finite size of the system means that once its present state stops being the global minimum of the free energy, it will be able to climb over the free-energy barrier between the local and global minima (which, as pointed out in Sec. III.C.1, would not be the case if $N \to \infty$, because the barrier rises extensively).

We notice the hysteresis, explained in Sec. IV.B, the difference between the rising $\tilde{\alpha}$ and the falling $\tilde{\alpha}$ results. However, because of the finite-size effects, the hysteresis may be avoided by allowing the algorithm a longer time to converge for each value of $\tilde{\alpha}$, which will mean that the system always reaches the global free-energy minimum, whether or not $\tilde{\alpha}$ is rising. This result is confirmed by Fig. 22(b), which shows numerical results with two different numbers of Monte Carlo steps allowed for the

stochastic dynamics. For more steps the first-order transition becomes sharper. The conclusion is that for finite-size systems, the transition between two states occurs at a value of $\tilde{\alpha}$ between the value of $\tilde{\alpha}$ at which the position of the global free-energy minimum changes and the value where the local minimum in which the system began disappears.

2. MLN learning nonlinearly separable rules

Two examples of MLN learning nonlinearly separable rules have so far been studied. Both examples were cases in which the teacher's network was identical to that of the student; i.e., the rule is learnable.

Schwarze and Hertz (1992a, 1992b) and Mato and Par-

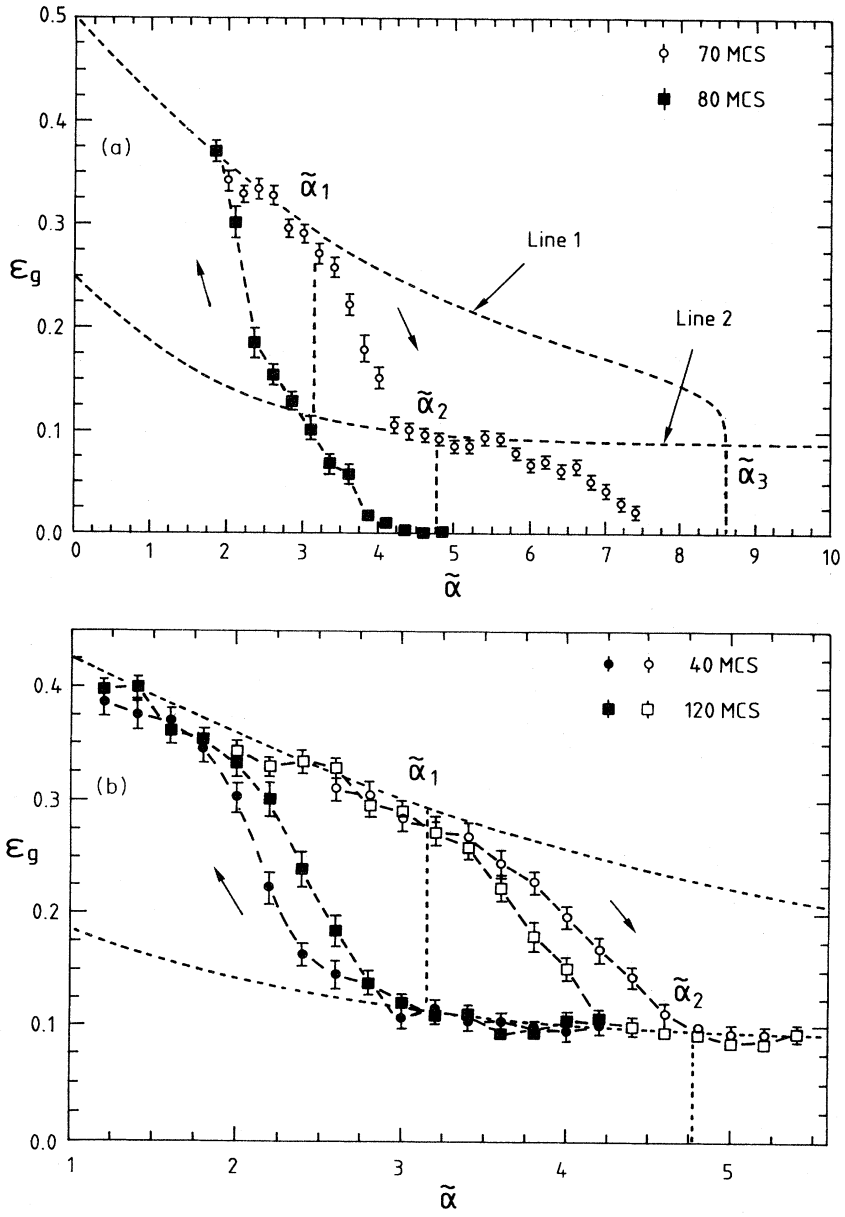

FIG. 22. The committee machine with $K=3$ and Ising couplings between the input layer and the hidden layer, learning a linearly separable rule at high temperature: (a) The dashed lines show the theoretical predictions explained in the text, but the simulation with $N=75$ was dominated by finite-size effects. Open symbols show the results with $\tilde{\alpha}$ rising, and the solid symbols show $\tilde{\alpha}$ falling. The number of Monte Carlo steps allowed per spin is marked on the top right corner. (b) An expanded section of the same diagram, but allowing the stochastic algorithm different numbers of steps.

ga (1992) analyzed a committee machine learning a committee machine, by which we mean learning a rule whose functional form is identical to that of a committee machine. Hansel *et al.* (1992) considered a parity machine learning a parity machine.

a. A committee machine learning a committee machine

Schwarze and Hertz (1992a) considered a committee machine with nonoverlapping receptive fields and a large number K of hidden units ($K \gg 1$, through still much less than N) learning at $T=0$ a rule generated by a network of identical architecture. Their analysis proves results similar to those of a spherical perceptron learning a linearly separable rule.

The order parameters have a meaning close to that of Sec. III. We use $q_k^{\gamma,\gamma'}=K\langle\langle \mathbf{J}_k^{\gamma}\cdot\mathbf{J}_k^{\gamma'}\rangle_{\beta}\rangle_{\xi}$, i.e., the (properly normalized) overlap of the N/K inputs to the kth hidden unit in two different replicas, and $R_k^{\gamma}=K\langle\langle \mathbf{J}_k^{\gamma}\cdot\mathbf{B}_k\rangle_{\beta}\rangle_{\xi}$, the overlap of the inputs to the kth hidden unit in the γth replica with the kth hidden unit of the teacher. Schwarze and Hertz (1992a) assume that all these parameters are independent of k.

Many of the formulas for the perceptron calculation of Sec. III.A carry through if one applies the following transformation to the order parameters q and R:

$$q\to\frac{2}{\pi}\sin^{-1}q,\quad R\to\frac{2}{\pi}\sin^{-1}R\ . \tag{6.15}$$

Let us, for example, work out the generalization function of the committee machine. A simple way to do this is by considering again the argument which gives us ϵ_f for a perceptron (defined by $\mathbf{J}$) learning a linearly separable rule (defined by $\mathbf{B}$; Sec. II.D). For a random question $\mathbf{S}$, we can define the variables $x_B\equiv\sqrt{N}\,\mathbf{B}\cdot\mathbf{S}$ and $x_J\equiv\sqrt{N}\,\mathbf{J}\cdot\mathbf{S}$. Since $\mathbf{J}$ and $\mathbf{B}$ are normalized and have overlap R, x_B and x_J are Gaussian distributed for random questions with $\langle x_B^2\rangle=\langle x_J^2\rangle=1$ and $\langle x_B x_J\rangle=R$. We showed in Sec. II.D by geometry that the generalization function, $\epsilon_f(\mathbf{J},\mathbf{B})=\langle\Theta(-x_B x_J)\rangle_s$, is equal to $(1/\pi)\cos^{-1}(R)$.

Returning now to the committee machine, we note, using the result for the perceptron, that the chance on a random question that the kth hidden unit of the student will disagree with the kth hidden unit of the teacher is $(1/\pi)\cos^{-1}(R_k)$. Let us assume symmetry of the hidden units, so that $R_k=R$ for all k. We can now define two quantities: Δ, which is $(1/\sqrt{K})$ times the sum of the hidden units of the teacher when question $\mathbf{S}$ is applied, and η, which is $(1/\sqrt{K})$ times the sum of the hidden units of the student. Since the committee machine has nonoverlapping receptive fields, both Δ and η are the sum of K independent terms. Thus by the central limit theorem they are Gaussian variables with $\langle\Delta^2\rangle=\langle\eta^2\rangle=1$, but they are also correlated so that $\langle\Delta\eta\rangle=(1/\pi)\cos^{-1}(R)\ -[1-(1/\pi)\cos^{-1}(R)]=(2/\pi)\sin^{-1}(R)$. The generalization function is $\langle\Theta(-\Delta\eta)\rangle$; therefore we can simply deduce, using the result of the previous paragraph, that for the committee machine with nonoverlapping receptive fields,

$$\epsilon_f(R)=\frac{1}{\pi}\cos^{-1}\left[\frac{2}{\pi}\sin^{-1}R\right]\ . \tag{6.16}$$

For spherical couplings the generalization error decreases for large α as $\epsilon_g\sim 1.25/\alpha$, which is twice the result of the perceptron. For Ising couplings the results from the perceptron carry over in a similar way. There is a thermodynamic transition to perfect generalization at $\alpha_{\rm th}(K\gg 1)$ forming the lower α bound for the existence of a large metastable region bounded from above by the spinodal $\alpha_{\rm sp}(K\gg 1)$.

It is remarkable that although $\alpha_{\rm th}(K\gg 1)<\alpha_{\rm th}(K=1)$, $\alpha_{\rm sp}(K\gg 1)>\alpha_{\rm sp}(K=1)$. As in the case of the Ising perceptron (Sec. III.C.1.b), the replica-symmetric spinodal line is incorrect, and RSB should be considered. However, this effect is likely to increase $\alpha_{\rm sp}$ still further.

Mato and Parga (1992) considered the same model as Schwarze and Hertz (1992a), but at high temperatures (Sec. III.A.5), which allowed them to extract results for an arbitrary number of hidden units K.

For $K\gg 1$ they found the same invariance property (6.15) as Schwarze and Hertz (1992a). The freezing transition has the same structure as the one outlined above. They showed that, in general, $\tilde{\alpha}_{\rm sp}(K)$ is an increasing and $\tilde{\alpha}_{\rm th}(K)$ a decreasing function of K. The existence of a freezing transition is supported by Monte Carlo simulations. They also verified numerically permutation symmetry of the order parameters.

Mato and Parga (1992) extended their analysis to committee machines with more than one hidden layer, for the special case in which the number of hidden units in the ith hidden layer K_i is much larger than the number of hidden units in the $(i+1)$th hidden layer K_{i+1} ($K_i\gg K_{i+1}$). They found for the freezing transition that $\tilde{\alpha}_{\rm th}(L)$ is a decreasing function of the number of layers L, bounded from below by $\tilde{\alpha}_{\rm th}(L\to\infty)\approx 1.38$. However, the spinodal point diverges like $\tilde{\alpha}_{\rm sp}(L)\sim(\pi/2)^L$ as $L\to\infty$.

Schwarze and Hertz (1992b) have also studied, for $K\gg 1$, a fully connected committee machine under the annealed approximation (Sec. III.A.6). For Ising weights they find that, as for the Ising binary perceptron, (Sec. III.C.1), there is a slow improvement in generalization followed by a first-order thermodynamic transition to perfect generalization at $\alpha=\alpha_{\rm th}$, which scales with K. However, unlike the Ising binary perceptron, the metastable solution persists for all α: no spinodal point is reached. As $\alpha\to\infty$, the generalization error of the metastable state tends to a nonzero value. In this model, the metastable state has unbroken committee symmetry (Sec. VI.B.2), while the student with perfect generalization, of course, has the connections to its hidden units aligned with those of the teacher.

Schwarze and Hertz (1992b) also consider continuous weights. For low α, the system adopts a committee symmetric phase. At a critical $\alpha=\alpha_{\rm th}$ there is a thermodynamic transition to a phase with broken committee symmetry, in which students have an overlap $\sim 1/K$. Once again, however, the metastable state persists for all α.

b. A parity machine learning a parity machine—memorization without generalization due to internal symmetries

In a different multilayer problem, Hansel *et al.* (1992) recently found results which have general implications. They were studying a parity machine with nonoverlapping receptive fields and two hidden units, learning at $T=0$ a rule generated by a network of identical architecture. From Eq. (6.4), we see that the model considered has a Z_2 symmetry; i.e., if we label the set of N connections into two hidden units by the single N-vector $\mathbf{J}$, the error associated with a network state $\mathbf{J}$ is the same as the one associated with $\mathbf{J}'=-\mathbf{J}$.

Assuming replica symmetry of the order parameters q and R, Hansel *et al.* (1992) found that up to a critical α_c memorization without generalization takes place, $\epsilon_g=\frac{1}{2}$. In this phase $\mathbf{J}$ and $\mathbf{J}'$ are in the same ergodic component. Above $\alpha=\alpha_*$ the saddle-point equations of the free energy have two solutions: one with $q=R=0$, which is unstable with respect to RSB, and a stable solution in which Z_2 symmetry breaks and $\mathbf{J}$ and $\mathbf{J}'$ cease to belong to the same ergodic component. As a result of this the generalization error of the stable phase starts to decrease from $\frac{1}{2}$ for $\alpha>\alpha_*$. Numerical simulations, performed using the least action algorithm, lend some support to the theory.

Retarded generalization due to internal symmetries had already been observed in simpler models, such as in the analysis of the unlearnable problem of a perceptron (student) trying to learn a parity machine (teacher); see Sec. V.A.3 and Watkin and Rau, 1992b), but the wider significance of the result was not realized. It can easily be interpreted using a Landau-Ginzburg argument (Hansel, private communication, 1992). Retarded generalization was also observed using the high-temperature approximation (Sec. III.A.5) for a parity machine with $K=2$ and Ising weights (Hansel *et al.*, 1992).

We may conjecture that retarded generalization occurs in any student space with an internal symmetry such that a network which is the mean of a set of networks related by the symmetry operation is uncorrelated with the teacher. Thus in the parity machine, the mean of $\mathbf{J}$ and $-\mathbf{J}$ is the null vector, whose overlap with the teacher vector $\mathbf{B}$ is zero. Notice that the committee symmetry of a fully connected committee machine does not fall into this category; so there is no contradiction with the work of Schwarze and Hertz (1992b) described above.

3. Overfitting

Neural networks have a very large number of free parameters which have to be adjusted. When the number of internal parameters is too large, the learning process results in overfitting, giving undue significance to individual examples. Since this is a real concern for machine learning with multilayer networks, and one which we feel deserves more investigation, we shall mention it here, although little statistical mechanics has so far been applied.

One of the key problems in the training of neural networks is that the complexity of the network needed to implement an unknown rule cannot, in general, be known in advance. If the architecture is too simple, the network will not be able to perform the desired task, whereas, as shown in Sec. VI.C, too complex a student leads to poor generalization (as well as inefficiency). Therefore a search for the optimal architecture should be part of a good learning strategy.

To avoid overfitting, one can, as mentioned above, try to reduce the number of superfluous hidden units or connections. Superfluous here means that their removal has the least effect on the training error. Algorithms based on the backpropagation of error have been proposed where unnecessary weights (Le Cun *et al.*, 1990) or neurons (Sietsma and Dow, 1988) are removed from the system. After each such change of architecture all the remaining neurons have to be retrained (this is termed a strong *nonlocality).* Of course, one problem with this method is that while removing superfluous parameters some important ones may also be removed; so the network would no longer possess enough flexibility.

Example: Utans and Moody (1991) recently presented an application of the methods described above to select the architecture of a neural network trained to predict the corporate bond rating on Wall Street. They used a two-layer network, whose number of hidden units they optimized by considering essentially the number of hidden units which minimizes the training error as a function of the number of hidden units of a network architecture which is otherwise fixed.

Having selected the number of hidden units, they proceed by reducing the number of weights using the method of *optimal brain damage* (Le Cun *et al.*, 1990). The quality of the prediction of this network exceeds by far that of previously existing methods, such as linear regression.

Another approach, well known from statistics, is *regularization.* In this case one does not minimize the training error in a gradient descent algorithm, but instead the training error plus a term which includes the absolute value of the weights so that gradient descent encourages the weights to decay to zero (e.g., Hinton, 1986; Hanson and Pratt, 1989). The size of the second term is parametrized by a constant η and a formula for choosing this parameter may be justified by a variety of arguments (MacKay, 1992; Moody, 1992). Alternatively η may be chosen by a process called "cross validation" (Hertz *et al.*, 1991).

Sjöberg and Ljung (1992) showed that performing a limited number of gradient descent iterations when searching for a minimum of the training error has the same effect as regularization. They also derived criteria

for when to stop the training process. Applying their method to the modeling of the dynamics of a hydraulically controlled robot arm, they could indeed show that the overfitting previously present in a gradient descent algorithm had been removed.

4. Edge and patch detectors

Recently Sympolinsky and Tishby (1990) studied the learning of a fundamentally different form of rule: one with an intrinsic dimensionality. The question is a string of $+1$ and -1 digits and the answer is the number of *domains* in the question. That is, how many lengths there are in which all digits are $+1$ and how many in which they are all -1.

This task may be performed by a committee machine whose architecture is given in Fig. 23, where a solid circle represents a neuron in state $+1$ and an open circle a neuron in state -1. The $N+1$ input nodes, $\{S_i\}$ $(i=0,\ldots,N)$, are set to be the digits of the binary question, and the $K=N$ hidden layer nodes, whose states are $\{S_k\}$, each receive inputs from the input nodes $i=k$ and $i=k-1$, with strengths of weights $J_{k,i=k}=+1$ and $J_{k,i=k-1}=-1$. The hidden layer nodes each perform synaptic emulation, Eq.(2.4), and have a threshold so that $S_k=\mathrm{sgn}(S_{i=k}-S_{i=k-1}-1)$, which implies that a hidden layer node k only has $S_k=1$ if $S_{i=k}$ is in a domain of $+1$ digits and $S_{i=k-1}$ is in a domain of -1 digits. Nodes of the hidden layer therefore just detect this kind of domain boundary. This is shown in Fig. 23, for $N=5$, where, for example, $S_{k=2}$ is $+1$ because $S_{i=1}=-1$ and $S_{i=2}=+1$. The output node o is linear: $S_o=(N+1+\sum_k S_k)$, which equals the total number of domains (although there may be an error of ± 1 at each end of the string, which can be avoided if we use cyclic boundary conditions in the input string, $S_{i=0}=S_{i=N}$).

The training from examples of a network with this architecture and with Ising couplings was analyzed using the annealed approximation (Sec. III.A.6), and the analysis may be simplified by noting that the problem has a relationship to the nearest-neighbor Ising model, whose solution is well known (Baxter, 1982). For all finite temperatures ($T>0$) the average generalization error for large α decreases exponentially quickly, $\epsilon_g \sim 2e^{-2\beta\alpha}$. It is worth noting that the annealed approximation, which is usually only valid at high temperatures, seems to provide results in agreement with Monte Carlo simulations for temperatures as low as $T\approx 0.3$. For temperatures $T<0.4$ the annealed approximation predicts a first-order transition in which ϵ_g drops to a low but finite value, which is in fact observed in Monte Carlo simulations.

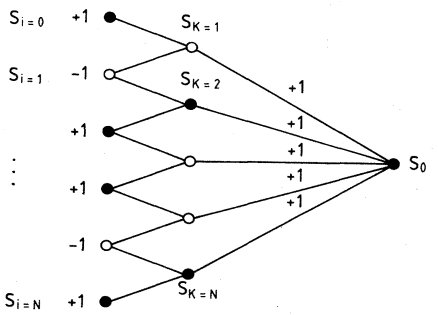

FIG. 23. A committee machine performing edge detection, as explained in the text. Solid circles indicate neurons in the $+1$ state and open ones are in the -1 state.

A related, well-known problem—the *contiguity* problem—is to identify the number of domains larger than a given threshold. It, too, may be solved, using a very similar architecture (Tishby *et al.*, 1989); Schwartz *et al.*, 1990; Sompolinsky and Tishby, 1990). The learning process is discontinuous for all temperatures and leads to perfect generalization beyond a certain α_c.

Kocher and Monasson (1991) have considered the related problem of counting the number of white patches on a black and white two-dimensional square lattice. One can easily construct a two-layer network with $2N^2$ neurons performing this task approximately. Within the annealed approximation the analysis of learning is substantially simplified by noting its relationship to the exactly solvable two-dimensional Ising model (Onsager, 1944; Baxter, 1982).

D. General behavior

In this section we describe two types of information applying to wide classes of network which can be extracted using statistics. The first relies upon the annealed approximation, but has often been used in practical applications. The second is exact prediction of asymptotic behavior.

In one of the seminal papers in the field, Levin *et al.* (1989) showed that within the annealed approximation, it is possible to write the generalization error which a system *would* have after correctly learning p examples of a rule, as a function only of quantities which can be *observed* for many fewer examples. In this way it has often been possible to tell whether learning with a network is a reasonable option (Solla, private communication, 1992).

What follows is a simplified version of the general argument, to which the reader is referred.

The generalization error we *would* obtain in learning a single rule V from p examples is the quantity $\langle\langle \epsilon_f(\mathcal{N},V)\rangle_{\mathcal{N}}\rangle_{\xi}$, where $\langle\cdots\rangle_{\mathcal{N}}$ again means the average over networks built using the training set by a certain algorithm. In this case we are not interested in learning how a "typical" rule does, and then relying upon self-averaging behavior. We want to know how *this* rule does; so we shall not average over $P(V)$.

For zero-temperature, zero-stability learning, we can write $\langle\langle \epsilon_f(\mathcal{N},V)\rangle_{\mathcal{N}}\rangle_{\xi}$ as

$$\left\langle \frac{\int d\mu(\mathcal{N})\epsilon_f(\mathcal{N},V)\prod_\mu \delta(\mathcal{N}(\xi^\mu),V(\xi^\mu))}{Z}\right\rangle_{\xi}, \quad (6.17)$$

where, as earlier, $\int d\mu(\mathcal{N})$ means the integration over

the measure of student space.

However, using the annealed approximation and noting that

$$\langle \delta(\mathcal{N}(\xi^\mu), V(\xi^\mu)) \rangle_\xi = 1 - \epsilon_f(\mathcal{N}, V) ,$$

we can approximate this as

$$\langle \langle \epsilon_f(\mathbf{J}, V) \rangle_{\mathcal{N}} \rangle_\xi \approx 1 - \frac{\int d\mu(\mathcal{N})(1-\epsilon_f(\mathbf{J}, V))^{p+1}}{\int d\mu(\mathcal{N})(1-\epsilon_f(\mathbf{J}, V))^p} . \tag{6.18}$$

To understand the meaning of this expression, let us introduce a function $\phi(g)$, which is just the proportion of student space having generalization function $1-g$. That is,

$$\phi(g) \equiv \int d\mu(\mathcal{N}) \delta(g - [1 - \epsilon_f(\mathcal{N}, V)]) .$$

Using this definition, it is thus clear that

$$\langle \langle \epsilon_f(\mathbf{J}, V) \rangle_{\mathcal{N}} \rangle_\xi \approx 1 - \frac{\int_0^1 dg\, g^{p+1} \phi(g)}{\int_0^1 dg\, g^p \phi(g)} , \tag{6.19}$$

which is the ratio of the pth and $(p+1)$th moment of $\phi(g)$.

The reason why this result is interesting is that $\phi(g)$ can be *observed,* at least approximately, using a small training set. We simply generate networks at random in the student space, work out the training error on a small training set, and use the definition of $\phi(g)$.

Despite the annealed approximation and the approximate nature of our knowledge of $\phi(g)$, this has been useful in several applications in order to predict how large a training set is required for successful generalization.

Levin *et al.* (1989) also pointed out that the asymptotic behavior for large p is determined by the value of $\phi(g)$ near $g=1$. If $\phi(g) \sim g^d$ for $g \to 1$, then $\epsilon_g \approx (d+1)/p$, which is $1/\alpha$ decay, as observed in Sec. III.C.1.

This consideration brings us to the second part of the section. What asymptotic behavior can be extracted from thermodynamics?

Seung, Sompolinsky, and Tishby (1992) gave results which apply to *any* "smooth" network, defined as one with continuous weights and in which $e(\mathcal{N}, V, \mathbf{S})$ is at least twice differentiable with respect to those weights. An example of such a network is the linear spherical perceptron (Sec. III.C.2). At large α, the free energy is dominated by the network (defined by $\mathbf{J}^*$) for which the generalization error is minimal, $\epsilon_g^{\min}$. For large, finite α, it is possible to expand around this solution. They introduced two matrices,

$$U_{ij} = \int d\mu(\mathbf{S}) \partial_i \partial_j e(\mathbf{J}^*, V, \mathbf{S}) \tag{6.20}$$

and

$$W_{ij} \equiv \int d\mu(\mathbf{S}) [\partial_i e(\mathbf{J}^*, V, \mathbf{S})][\partial_j e(\mathbf{J}^*, V, \mathbf{S})] , \tag{6.21}$$

where ∂_i means the derivative with respect to the ith component of $\mathbf{J}$.

Seung, Sompolinsky, and Tishby (1992) can then show that, provided a stochastic algorithm converges at all, learning by minimizing $E_t(\mathcal{N})$ at a temperature T gives

$$\epsilon_g(T, \alpha) = \epsilon_g^{\min} + \left[T + \frac{1}{N} \mathrm{Tr} W U^{-1} \right] \frac{1}{2\alpha} ,$$

$$\epsilon_t(T, \alpha) = \epsilon_g^{\min} + \left[-\frac{1}{N} \mathrm{Tr} W U^{-1} \right] \frac{1}{2\alpha} ,$$

with a correction of order $1/\alpha^2$ in both cases. In the special case of a learnable rule, $\epsilon_g^{\min} = 0$; and it is also possible to show that $U_{ij} = 0$. Then

$$\epsilon_g^{\min} = \frac{T}{2\alpha} + O\left[\frac{1}{\alpha^2} \right] . \tag{6.22}$$

At zero temperature, the leading term disappears, as do all higher terms, implying that as α rises there must be a transition to perfect learning.

Amari and Murata (1991) have been able to show a similar result for a different class of networks using conventional statistical techniques. However, their analysis only applies to stochastic networks in which the output is not a deterministic function of the input.

E. Constructive algorithms

A different class of training algorithms actually constructs the network while learning, with the simple goal of producing a network with zero training error (Gallant, 1986). Very little statistical mechanics has been performed on constructive algorithms, but we have included a brief summary here because it is a field of considerable practical importance.

Rather than removing degrees of freedom from too complicated a structure, constructive algorithms add neurons or layers of neurons if a given network fails to learn the training set. This has to be one such that the representations of the training patterns remain or become faithful. Learning is performed locally for the most recently added neuron by use of perceptron algorithms. Convergence is guaranteed if the input values are restricted to $+1$ or -1; that is, any binary classification can be implemented of a set of binary input patterns, and most of the procedures (and the convergence proofs) can be generalized to bounded, continuous input variables. However, the number of neurons needed for specific problems may be very large (in the worst case rising in proportion to the number of examples in the training set).

The *tiling algorithm* by Mézard and Nadal (1989), for example, first tries to realize the correct mapping with a single *master neuron,* shown as neuron D in Fig. 24(a). If this attempt fails because the problem is not yet linearly separable, so-called ancillary units, such as neurons E and F, are added to this layer [cf. Fig. 24(b)] until the internal representation becomes faithful, which requires a lower number of neurons than the total in the first layer. The whole process is then repeated, starting with master

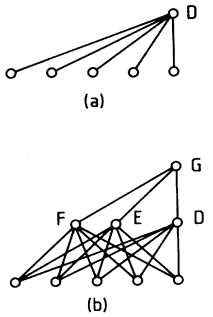

FIG. 24. Constructing a network: (a) The hidden layer's master neuron D receives inputs from all neurons in the input layer. (b) Nodes E and F are added to the second layer until the representation is faithful. G is the master neuron of the third layer.

neuron G in the third layer, with the first hidden layer taking the place of the input layer. The process continues until a layer is reached in which the master neuron is able to correctly classify all the patterns. The problem has then been solved.

Nadal (1989) studied a variation of this procedure in which *only* master neurons are added—connected to the input layer and all previous master neurons. In this case the search for a faithful representation can be omitted, since each master neuron is connected to the inputs itself.

By contrast, *only* ancillary neurons are added to the *neural tree* suggested by Sirat and Nadal (1990). This network has the advantage that it is not necessary for every neuron in the network to perform its function: the network's answer to a certain input is determined by propagation down a binary decision tree according to the states of some neurons, and other neurons need not be consulted. Frean (1990) independently proposed the *upstart* algorithm, which constructs a hierarchically organized network. In fact, this architecture can be mapped into the neural tree and vice versa (Hertz *et al.*, 1991).

Marchand *et al.* (1990) considered a network with only one hidden layer. Neurons are added one by one within this layer until it is certain that the output neuron only has to perform a linearly separable function to correctly classify all examples in the training set.

Biehl and Opper (1991), and independently Martinez and Esteve (1992), introduced the concept of tilinglike learning using a parity machine (Sec. VI.A). The growth of the network is restricted to adding neurons to the one hidden layer, and the most recently added neuron is trained to correct as many as possible of the errors the machine made before it was added. Again the number of errors decreases successively.

Many other construction schemes, which are beyond the scope of this review, exist [see, for example, Bichsel and Seitz (1989); Ruján and Marchand (1989); Fahlman and Lebiere (1990); Golea and Marchand (1990); Knerr *et al.* (1990); and Zollner *et al.* (1992)].

VII. DISCUSSION AND OUTLOOK

In this section we first discuss the usefulness of the statistical-mechanics theory of learning a rule which we have developed in this paper. We present another line of research which seems to us to require a deeper understanding, and we suggest directions in which we expect future research to lead.

A. The value of a statistical-mechanics approach to learning

The great interest during the past few years in the statistical theory of learning a rule may be motivated from three different points of view: in terms of its usefulness in understanding mathematical tools used frequently in physics; as a solvable instance of the very common problem of inference from noisy data; and, of course, in order to advance understanding of neural networks. We shall treat these topics in order.

A good example of the first point—the usefulness of this theory in understanding the tools of physics—was discussed in Sec. IV.B. Learning in the discrete student space of an Ising binary perceptron has been solved using a dynamic mean-field theory and using the method of replicas, and although both methods are "believed to be exact," the results are contradictory. The obvious explanation is that the full meaning of the method of replicas is still to be discovered: either some principle of RSB is lacking, or, more interestingly, this model may demonstrate that replica symmetry breaking is not equivalent to ergodicity breaking.

Learning a rule is just one example of inferring underlying structure (in this case a rule) from "experimental data" (the training set). A second justification of the statistical theory of learning is therefore to introduce the advanced methods of statistical mechanics with a quenched disorder into the important field of statistical inference. The same questions we can answer for neural networks may be significant elsewhere: How much can be expected of noisy data? Can one discover, without writing a long program, how long a reconstruction algorithm will take to run? How should the reconstruction be performed in principle?

Of course, one reason why neural networks have been so easily tackled using statistical mechanics is that their high connectivity implies that mean-field theory is exact. Many other inference problems have a much lower dimensionality. For example, lines are fitted from data values measured along a one-dimensional axis. Images often have to be reconstructed from blurred and noisy data, but images are intrinsically two dimensional. Exact solution of inference problems might be of interest, but it is hard to see how practical use might be made of the sort of information which can be extracted from low dimensional statistical physics (critical exponents, etc.). This restricts the usefulness of statistical mechanics to unusual

inference problems in which, for some reason, models may be solved. One example of this is calculating the efficiency of *coding* algorithms for information transmission (Sourlas, 1989).

The main point of neural network theory remains, of course, understanding neural networks. Statistical mechanics provides a natural formalism within which disparate approaches may be connected, from simple backpropagation to sophisticated Bayesian techniques.

Statistical mechanics differs from the other theory of network learning, VC theory, in that a specific form is assumed for the network, for the distribution of examples, and for the prior probability of rules, $P(V)$. Within these assumptions exact predictions for the success of learning can often be made.

In advanced engineering problems, such as speech recognition, what would $P(V)$ be, and how could we possibly know it? One obvious claim is that in the limit in which we have no prior information about V, $P(V)$ is constant over all the space of Boolean functions which could connect questions and answers. The version space consists of all Boolean functions which agree with the examples we learned from. In this limit, of course, $P(V)$ does not correlate the answers to questions of the training set with the answer to different questions, and so the expectation of our ability to generalize should be zero.

However, neural networks *are* successful in practice, applied to problems such as speech recognition, where correlation between examples is expected to occur. $P(V)$ must therefore be biased towards those rules with "more regularity" than the random Boolean function—it in some sense encodes our intuition that the problem we are trying to learn is a relatively "simple" classification. Recently attempts have been made to rigorize the rather diffuse concept of a function's "complexity"; but it is still far from obvious how to proceed and this remains an important project for the future.

The fact that the predictions made by the statistical theory of learning depend upon $P(V)$ may be turned to advantage, however. Consider, for example, learning a rule with a two-layer network of a certain size. We might believe that this network is complex enough to learn the rule perfectly—that is, we make an assumption that $P(V)$ is only nonzero for rules which can be written in the same functional form as the network. Having made this assumption, we can compare the results observed in practice to predictions made using statistical mechanics. If there are departures from the theoretical prediction, this demonstrates that our assumption about $P(V)$ was incorrect, and to learn the rule perfectly we shall need a more complex network. Presumably the size of the discrepancy between theory and experiment is a measure of the inaccuracy in our estimation of $P(V)$, and thus an estimate of how much more complex our network must become. This sort of test, in which a model (in this case for V) is rejected without comparison with a different model, is called an alternative-free test. It has the advantage that we do not need to know the true $P(V)$ in order to apply it.

An alternative justification for the statistical theory of learning a rule is as a means of gaining an insight into a field which is famously hard to rigorize, and so improve learning algorithms. An obvious example is how statistical mechanics lets us design the cost function to be minimized, for example, by simulated annealing or backpropagation. In the noisy prototype problem, for example, we find that a smooth energy function is far more efficient than a discontinuous one, since it reduces the liability of overfitting the training set.

Another justification for the theory might be in determining how much information can *in principle* be extracted from a given amount of data. Suppose again that we are learning a certain rule, and we have devised a simple algorithm to do this, for example, just a minimization of the training error by backpropagation. As we have argued above, if a statistical prediction agrees with the experimental results, then this is good evidence for the "complexity" of the rule; and so a curve can be produced theoretically showing the success of the optimal algorithm. By the discrepancy between the generalization error our simple algorithm produces and the minimum generalization error, we can judge how worthwhile it is to search for a *better* learning algorithm. For example, as we have seen, maximum stability learning of a linearly separable rule gives the generalization ability shown in Fig. 10 as line 2, while the optimal result is line 3. Only if it is really worthwhile, in a given engineering situation, to obtain the result of line 3 instead of line 2 is it worthwhile to search for a better algorithm than maximum stability learning [happily the contrast between simple algorithms and optimal ones is greater for more difficult problems (see Sec. V.A.4)].

Lastly, we have seen that it is possible to gain useful information from statistical mechanics by relating unknown but interesting quantities to observable ones (Sec. VI.D).

B. The complexity of a rule

Here we shall mention a fundamental but relatively unexplored avenue of research which has been suggested by physicists, but does not quite fit into the main statistical theory of learning a rule.

Carnevali and Patarnello (1987) considered all the $2^{16}=65\,536$ Boolean functions, that is, rules, which may be performed on four two-state inputs and all the 4×10^{10} possible networks which may be constructed out of four logic gates, each of which performs any logical function of just two inputs. They counted the number of networks performing each function and found that slightly different Boolean functions are performed by enormously different numbers of networks. Van den Broeck and Kawai (1990) obtained similar results by randomly constructing 10^7 networks out of up to 200 logic gates and checking which Boolean function each performs. They found that the probability $P(\zeta)$ that a randomly chosen Boolean function will be implemented by ζ networks is

$$P(\xi) \propto \xi^{-\alpha} \tag{7.1}$$

with $\alpha \approx 0.7$, which they took as evidence for an underlying fractal structure for the space of Boolean networks. The work of Parisi and Slanina (1992) suggests that important features such as the ease of generalization may depend only upon rules, and not upon the choice of architecture we have made. This suggests the possibility of defining the *complexity* of a rule in a way independent of the network architecture. Although some of these papers are formulated in terms of thermodynamics, little analytical work has been performed and the most important questions remain unanswered.

C. Outlook

This article has argued that statistical mechanics is a powerful alternative to VC theory, but we believe that an important topic of new research will be to reconcile the two approaches. For example, recently attempts have been made to evaluate the VC dimension of a multilayer network by comparing how it learns two different training sets (Levin *et al.*, 1992; Vapnik, 1992). This sounds like a problem it would be possible to tackle using statistical mechanics, by calculating fluctuations in measurable quantities for different realizations of the quenched disorder.

The most fundamental open question in the field concerns an understanding of the mappings multilayer networks make between the spaces of questions and answers. In Sec. VI, we presented the early attempts to introduce statistical mechanics into the problem, but clearly our intuition is still very poor.

Instead of dividing input space by a hyperplane, multilayer networks make a much more complex partition, shown schematically in Fig. 25(a). The + and − signs represent the answers to given questions, which a certain network answers correctly by giving the answer −1 to all questions in the shaded areas. In this case the partition has produced disconnected areas in input space, but clearly the topology of the partition is not unique; the partition of Fig. 25(b) also learns all the training set perfectly. For good generalization, the partition must resemble the one made by the rule, and our intuition would prefer the network we design to make as *smooth* a partition as possible [Fig. 25(a) rather than Fig. 25(b)]. This is in accordance with the common lore of inverse problems (Karl, 1989). An understanding of the relationship between an architecture and the topology of partition it produces may give us a more powerful criterion for the construction of a multilayer network than simply ensuring that it produce a training error of zero. It may well be worth adding more neurons with more constraints to obtain student spaces with the correct topological freedoms. This approach would also be useful in incorporating prior information about the topology of the rule.

In Sec. VI.B.3 we discussed attempts to relate RSB to the failure of backpropagation schemes. As we showed,

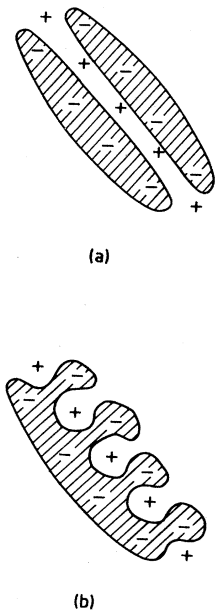

FIG. 25. Schematic diagrams of two partitions of the question space which correctly answer all the training set. The shaded area in each diagram is that to which a possible network would answer −1, and the training set in each case is illustrated here by the + and − signs. In (a) the partition is simple, but disconnected. In (b) the partition is connected, but complicated.

there are still important open questions. Is backpropagation able to converge in RSB regions, or is RSB very dependent upon the smoothness of the energy landscape? Answering these questions would help us to design training energies to ensure convergence of training algorithms.

The problems we have chosen to discuss in this paper seem to us to capture features of realistic network programming. However, all too few have been studied as part of a project to solve a real problem. We believe that such work is important, forcing us to incorporate features of real learning problems, at least approximately.

Several applications of neural networks have been hardly touched upon in this review, because there is almost no statistical mechanics to describe them. The most important is the *modeling of chaotic time series,* such as the prediction of stock market prices. Neural networks are all too often treated as "black boxes" in such applications, so that we are unable to quantify the reliability of their predictions (except by observation), nor to improve them systematically.

In Sec. VI.E we mentioned that rather than considering the layers already constructed to be fixed, we could perform a stochastic growth algorithm using an energy cost function incorporating the appropriate definition of "simple"—for example, the number of nodes—so that the network could grow and shrink spontaneously. Quenched noise would arise from the examples we are learning from (again) and the distribution from which the rule to be learned is chosen. The method of replicas has

been successful in the past (Gardner *et al.*, 1989) at showing when the student space becomes disconnected, which would give us some insight into the spurious training energy minima to which, as pointed out in Sec. VI.B, multilayer networks are subject, and perhaps even a clue for avoiding them.

From a physical point of view, we can consider the construction of multilayer networks as an *aggregation* problem; new neurons are added to an enlarging structure. The statistical mechanics of some aggregating systems has recently been investigated (for example, Derrida, Hakim, and Vannimenus, 1991) and shows a strong non-self-averaging dependence of the final result upon the initial pattern of growth. In growing a network, there are two differences: there is a rule to be learned after which growth ceases and therefore there are a (presumably infinite) number of fixed points for the dynamics; also, according to the learning algorithm used, the energy function governing growth may be complex and long range.

VIII. CONCLUSION

We have shown that advanced statistical mechanics is a valuable tool in the analysis of learning rules with neural networks, and conversely that new insights have been generated for use in conventional physics. The problem is especially interesting as a class of strongly interacting complex systems, many of whose properties may be solved *exactly* in the thermodynamic limit.

The value of statistical mechanics derives from its success in analyzing the self-averaging quantities of interest in the presence of quenched disorder: the examples we must learn from. As we have seen, this allows a novel treatment of information-theoretic inference. This surely has implications in many situations where physicists deduce underlying structures from experimental data sets corrupted by quenched noise. Predictions should be optimally *placed* within the version space of theories consistent with the data—the *hypothesis space* (Jaynes, 1983).

Networks learning rules are closely related to spin glasses: examples to be learned correspond to pairs of spins which must, or must not, be aligned. In both disordered systems it is the free energy which is the self-averaging quantity, and its average may be calculated using the method of replicas. The ergodicity breaking of spin glasses gave rise to anomalously long relaxation times, which first made them of interest for physicists; in neural networks, the same effect implies that the space in which networks evolve under a stochastic dynamics may become disconnected, leading to exponentially long learning times. Replica symmetry breaking in a spin glass is associated with frustration—it may be impossible to align all the spins so that all bonds are satisfied. Similarly, a neural network may reach a state in which all examples have not been learned and yet any small perturbation of the network within the student space leads to a larger training error; many valleys of the energy landscape exist around points which get different selections of the training set right. This is despite the fact that, for learnable rules, a network with zero training error does exist in a different area of student space where ergodicity is unbroken.

Luckily this strange behavior disappears when we have more examples to learn from and, for networks of continuous weights, often does not occur at all. Having seen in Sec. V.F that it is possible to choose the energy function so as to sculpt the energy surface, it is important to know the extent to which the slowing of the dynamics may be reduced, a question with important implications for other optimization problems.

The network of Sec. IV.A is a rare example of a strongly interacting system whose approach to the minimum of a noisy energy landscape can be analyzed exactly from a knowledge of the eigenvalue spectrum of a noisy matrix. Inspired by the question of the previous paragraph, could we choose our energy function here to reduce (or avoid) the critical slowing down explained in Sec. IV.A?

Frustration will always occur of course, if the rule is unlearnable: the student space does not contain a network which reproduces the rule. Since practical problems are expected always to fall into this category, it is important to understand how these affect the picture presented in Sec. III. High-temperature learning of the rule in Sec. V.A.2, for example, suggests that when Ising perceptrons learn unlearnable problems, their first-order transition is at first only into the region of minimal generalization error. What is the zero-temperature analog of this result and what is its interpretation? Does it change if optimal learning is employed, incorporating our knowledge of the form of the rule?

The question remains of whether any of the learning schemes which have some biological counterpart, although it seems that all but the simplest (Sec. II.D) involve nonlocal learning rules (one component of the weight vector is set according to a complex function of other components of the questions), which is said to be biologically implausible (Amit, 1989). A deeper insight into the growth of real networks, however, may lead to new classes of learning schemes.

Finally, we point out again that the next breakthrough in learning can be expected in multilayer networks, which are already in common engineering use. Here there are two distinct questions: What is the minimal size of network required to learn a rule? How should such a network be taught from examples? An answer to either question seems to require an insight into the mappings multilayer networks make between their input and output spaces which does not yet exist. Analysis of learning complicated rules drawn from some distribution again requires statistical mechanics, as will the analysis of stochastic multilayer algorithms, with networks growing and shrinking to meet the demands of new examples. The high degeneracy in the number of networks which can reproduce a given rule exactly may cause problems in

finding the minimal architecture, as will spurious solutions with a low but nonzero generalization error.

The learning of rules by neural networks is a fascinating ongoing problem, with cross-disciplinary implications for conventional physics, information theory, biology, and engineering. It is an excellent area in which to apply exactly the techniques of statistical mechanics and in which to develop new ones, in order to gain some insight into other strongly interacting complex systems.

ACKNOWLEDGMENTS

We acknowledge very fruitful discussions with Manfred Opper, Holm Schwarze, and Sebastian Seung. We thank Andreas Wendemuth and Dominic O'Kane for advice on presentation, and Marc Bouten and Roger Serneels for pointing out many places in which our test could be improved. We are particularly grateful to David Sherrington and Wolfgang Kinzel for their contributions to the evolving manuscript.

Many parts of this article were revised and updated during a visit to the Weizmann Institute of Science, Israel. Two of us (T.L.H.W. and A.R.) would like to thank the Electronics Department for their kind hospitality. We would also like to thank Eytan Domany, Ido Kanter, Ronny Meir, Tal Grossman, Yoav Freund, Eldad Barkai, and David Hansel for many fruitful discussions and especially for sharing their knowledge of multilayer networks.

One of us (T.L.H.W.) is grateful for the advice and encouragement of Naftali Tishby and Sara Solla, given during a recent summer school organized for NATO by Jean-Pierre Nadal, and to David MacKay for some more recent suggestions.

We thank the Hong Kong University of Science and Technology and especially Kwok-Yee Michael Wong for their support while the finishing touches were applied to this article.

This work was largely supported by the Science and Engineering Research Council of Great Britain. A.R. would like to thank the Studienstiftung des deutschen Volkes and Corpus Christi College, Oxford, for the award of two scholarships. He also acknowledges the award of the MINERVA prize of the Max-Planck-Gesellschaft. M.B. was supported by the Volkswagen Stiftung. T.L.H.W. thanks St. John's College, Cambridge for the award of a research fellowship.

REFERENCES

Abu-Mostafa, Y.S., 1989, Neural Comput. **1**, 312.

Ackley, D. H., G. E. Hinton, and T.J. Sejnowski, 1985, Cognitive Sci. **9**, 147.

Amari, S.-I., and N. Murata, 1991, "Statistical theory of learning curves under entropic loss criterion," University of Tokyo preprint.

Amit, D. J., 1989, *Modeling Brain Function* (Cambridge University, Cambridge, England).

Amit, D. J., H. Gutfreund, and H. Sompolinsky, 1985, Phys. Rev. Lett. **55**, 1530.

Anderson, P. W., Phys. Today, "Reference Frame," Jan. 1988, March 1988, June, 1988, Sept. 1988, July 1989, Sept. 1989, and March 1990.

Anlauf, J. K., and M. Biehl, 1989, Europhys. Lett. **10**, 687.

Anlauf, J. K., and P. Kuhlmann, 1992, "Metastable states in the projection rule network," Universität Giessen preprint.

Barkai, E., D. Hansel, and I. Kanter, 1990, Phys. Rev. Lett. **65**, 2312.

Barkai, E., D. Hansel, and H. Sompolinsky, 1992, Phys. Rev. A **45**, 4146.

Baum, E. B., 1991, IEEE Trans. Neural Networks **2**, 5.

Baum, E. B., and D. Haussler, 1989, Neural Comput. **1**, 151.

Baxter, R. J., 1982, *Exactly Solved Models in Statistical Mechanics* (Academic, London).

Bex, G.-J., 1992, private communication.

Bichsel, M., and P. Seitz, 1989, Neural Networks **2**, 133.

Biehl, M., 1989, Diplomarbeit (Universität Giessen).

Biehl, M., J. K. Anlauf, and W. Kinzel, 1991, in *Neurodynamics,* edited by F. Pasemann and H. D. Doebner (World Scientific, Singapore), p. 194.

Biehl, M., and M. Opper, 1991, Phys. Rev. A **44**, 6888.

Biehl, M., and H. Schwarze, 1992, Europhys. Lett. **20**, 733.

Binder, K., and D. W. Heerman, 1988, Monte Carlo Simulations in Statistical Mechanics (Springer, Berlin).

Binder, K., and A. P. Young, 1986, Rev. Mod. Phys. **58**, 801.

Bohr, H., J. Bohr, S. Brunak, R. J. M. Cotterill, H. Fredholm, B. Lautrup, and S. B. Peterson, 1990, FEBS Lett. **261**, 43.

Bollé, D., and P. Dupont, 1990, in *Statistical Mechanics of Neural Networks,* edited by L. Garrido (Springer, Berlin).

Bollé, D., P. Dupont, and J. Huyghebaert, 1992, Phys. Rev. A **45**, 4194.

Bollé, D., P. Dupont, and J. van Mourik, 1991, J. Phys. A **24**, 1065.

Bronstein, I. N., and K. A. Semendjajew, 1985, *Taschenbuch der Mathematik* (Harri Deutsch, Thun), p. 677.

Brout, R., 1965, *Phase Transitions* (Benjamin, New York).

Bryson, A. E., and Y.-C. Ho, 1969, *Applied Optimal Control* (Blaisdell, New York).

Carnevali, P., and S. Patarnello, 1987, Europhys. Lett. **4**, 1199.

Chesterton, G. K., 1910, "What's Wrong with the World," Chap. IV, p. 14.

Cottrell, G. W., P. Munro, and D. Zipser, 1987, in *9th Annual Conference of the Cognitive Science Society* (Erlbaum, Hillsdale, NJ), p. 462.

Cover, T.M., 1965, IEEE Trans. Electron. Comput. **EC-14**, 326.

de Almeida, J.R.L., and D. J.Thouless, 1978, J. Phys. A. Math. Nucl. Gen. **11**, 983.

De Dominicis, C., 1978, Phys. Rev. B **18**, 4913.

De Figueiredo, R.J. P., 1980, J. Math. Anal. Appl. **38**, 1227.

Del Giudice, P., S., Franz, and M. A. Virasoro, 1989, J. Phys. (Paris) **50**, 121.

Denker, J. S., D. Schwartz, B. Wittner, S. A. Solla, R. E .Howard, L. D. Jackel, and J. J. Hopfield, 1987, Complex Syst. **1**, 877.

Derrida, B., 1981, Phys. Rev. B **24**, 2613.

Derrida, B., E. Gardner, and A. Zippelius, 1987, Europhys. Lett. **4**, 167.

Derrida, B., R. B. Griffiths, and A. Prügel-Bennett, 1991, J. Phys. A **24**, 4907.

Derrida, B., V. Hakim, and J. Vannimenus, 1991, Phys. Rev. Lett. **43**, 888.

Derrida, B., and J. P. Nadal, 1987, J. Stat. Phys. **49**, 993.
Diederich, S., and M. Opper, 1987, Phys. Rev. Lett. **58**, 949.
Domany, E., W. Kinzel, and R. Meir, 1986, Europhys. Lett. **2**, 175.
Domany, E., J. L. van Hemmen, and K. Schulten, 1991, Eds. *Models of Neural Networks* (Springer, Berlin).
Eckmiller, R., 1989, in *Neural Computing Architectures. The Design of Brainlike Machines,* edited by I. Aleksander (North Oxford Academic, London), p. 305.
Edwards, S. F., 1970, in *4th International Conference on Amorphous Materials,* edited by R. W. Douglas and B. Ellis (Wiley, New York).
Edwards, S. F., and P. W. Anderson, 1975, J. Phys. F **5**, 965.
Edwards, S. F., and R. C.Jones, 1976, J. Phys. A **9**, 1595.
Eisenstein, E., and I. Kanter, 1992, "Numerical study of backpropagation learning algorithms for multilayer networks," Bar-Ilan University preprint.
Engel, A., H. M. Köhler, F. Tschepke, H. Vollmayr, and A. Zippelius, 1992, Phys. Rev. A **45**, 7590.
Fahlmann, S. E., 1989, in *Proceedings of the 1988 Connectionist Models Summer School,* edited by D. Touretzky, G. Hinton, and T. Sejnowski (Morgan Kaufmann, San Mateo), p. 38.
Fahlman, S. E., and C. Lebiere, 1990, in *Advances in Neural Information Processing Systems II,* edited by D. S. Touretzky (Morgan Kaufmann, San Mateo), p. 524.
Falk, H., 1975, J. Phys. C **8**, L294.
Fisher, M. E., and D. S. Gaunt, 1964, Phys. Rev. A **133**, 224.
Frean, M., 1990, Neural Comput. **2**, 198.
Gallant, S. I., 1986, in *Proceedings of the Eighth Annual Conference of the Cognitive Science Society* (Erlbaum, Hillsdale, NJ), p. 652.
Gardner, E., 1986, J. Phys. A **19**, L1047.
Gardner, E., 1987, Europhys. Lett. **4**, 481.
Gardner, E., 1988, J. Phys. A **21**, 257.
Gardner, E., and B. Derrida, 1988, J. Phys. A **21**, 271.
Gardner, E., and B. Derrida, 1989, J. Phys. A **22**, 1983.
Gardner, E., I. Yekutieli, and H. Gutfreund, 1989, J. Phys. A **22**, 1995.
Garey, M. R., and D. S. Johnson, 1979, *Computers and Intractability: A Guide to NP-Completeness* (Freeman, New York).
Geszti, T., 1990, *Physical Models of Neural Networks* (World Scientific, Singapore).
Golea, M., and M. Marchand, 1990, Europhys. Lett. **12**, 205.
Gross, D. J., and M. Mézard, 1984, Nucl. Phys. B **240**, 431.
Grossman, T., 1990, in *Advances in Neural Information Processing Systems II,* edited by D. S. Touretzky (Morgan Kaufmann, San Mateo), p. 516.
Grossman, T., and E. Domany, 1992, private communication.
Grossman, T., R. Meir, and E. Domany, 1988, Complex Syst. **2**, 555.
Gutfreund, H., and G. Toulouse, 1992, "The physics of neural networks," in *Spin Glasses and Biology,* edited by D. L. Stein (World Scientific, Singapore).
Györgyi, G., 1990a, Phys. Rev. Lett. **64**, 2957.
Györgyi, G., 1990b, Phys. Rev. A **41**, 7097.
Györgyi, G., and N. Tishby, 1990, in *Neural Networks and Spin Glasses,* edited by W. K. Theumann and R. Köberle (World Scientific, Singapore), p. 3.
Hammann, J., M. Ocio, E. Vincent, and R. Orbach, 1992, Physica A **185**, 278.
Hansel, D., G. Mato, and C. Meunier, 1992, Europhys. Lett. **20**, 471.
Hansel, D., 1992, private communication.
Hansel, D., and H. Sompolinsky, 1990, Europhys. Lett. **11**, 687.
Hanson, S. J., and L. Pratt, 1989, in *Advances in Neural Information Processing Systems I,* edited by D.S. Touretzky (Morgan Kaufmann, San Mateo), p. 117.
Hebb, D. O., 1949, *The Organization of Behavior* (Wiley, New York).
Hecht-Nielsen, R., 1987, in *IEEE 1st International Conference on Neural Networks,* Vol. III, edited by M. Caudill and C. Butler (IEEE, New York). p. 455.
Helmbold, D. P., and P.M. Long, 1991, in *Proceedings of the Fourth Annual Workshop on Computational Learning Theory,* edited by M. K. Warmuth and L. G. Valliant (Morgan Kaufmann, San Mateo).
Hertz, J. A., A. Krogh, and R. G. Palmer, 1991, *Introduction to the Theory of Neural Computation* (Addison-Wesley, Redwood City, CA).
Hertz, J. A., A.Krogh, and G. I. Thorbergsson, 1989, J. Phys. A **22**, 2133.
Hinton, G. E., 1986, in *Proceedings of the 8th Annual Conference of the Cognitive Science Society* (Erlbaum, Hillsdale, NJ), p. 1.
Hopfield, J. J., 1982, Proc. Natl. Acad. Sci. U.S.A. **79**, 2554.
Horner, H., 1992a, Z. Phys. B **86**, 291.
Horner, H., 1992b, Z. Phys. B **87**, 371.
Huang, K., 1987, *Statistical Mechanics* (Wiley, New York), p. 427.
Ising, E., 1925, Z. Phys. **31**, 253.
Itzykson, C., and J.-M. Drouffe, 1989, *Statistical Field Theory II* (Cambridge University, Cambridge, England).
Jacobs, R. A., 1988, Neural Networks **1**, 295.
Jaynes, E. T., 1957, Phys. Rev. **106**, 620.
Jaynes, E.T., 1983, *Papers on Probability, Statistics and Statistical Physics,* Synthese Library Vol. 158, edited by R. D. Rosenkranz (Reidel, Dordrecht).
Jeffreys, H., 1939, *Theory of Probability* (Clarendon, Oxford).
Kac, M., 1968, in *Trondheim Theoretical Physics Seminar,* Nordita Publ. No. 286.
Kanter, I., 1988, Phys. Rev. A **37**, 2739.
Kanter, I., 1992, private communication.
Karl, J. H., 1989, *An Introduction to Digital Signal Processing* (Academic, San Diego).
Kinouchi, O., and N. Caticha, 1992, "Optimal generalization in perceptrons," Universade de São Paulo preprint.
Kinzel, W., and M. Opper, 1991, in *Models of Neural Networks,* edited by E. Domany, J. L. van Hemmen, and K. Schulten (Springer, Berlin), p. 149.
Kinzel, W,. and P. Ruján, 1990, Europhys. Lett. **13**, 473.
Knerr, S., L. Personnaz, and G. Dreyfus, 1990, in *Neurocomputing: Algorithms, Architectures and Applications* (NATO ASI Series) (Springer, Berlin).
Kocher, I., and R.Monasson, 1991, Int. J. of Neural Syst. **2**, 115.
Köhler, H., S. Diederich, W. Kinzel, and M. Opper, 1990, Z. Phys. B **78**, 333.
Kohonen, T., 1988, *Neural Network Architectures* (Kogan Page, London).
Kramer, A. H., and A. Sangiovanni-Vincentelli, 1989, in *Advances in Neural Information Processing Systems I,* edited by D. S. Touretzky (Morgan Kaufmann, San Mateo).
Krauth, W., and M. Mézard, 1987, J. Phys. A **20**, L745.
Krauth, W., and M. Mézard, 1989, J. Phys. (Paris) **50**, 3057.
Krogh, A., 1992, J. Phys. A **25**, 1119.
Krogh, A., and J. A. Hertz, 1991, in *Advances in Neural Information Processing Systems III,* edited by D. S. Touretzky (Morgan Kaufmann, San Mateo).

Krogh, A., G.I. Thorbergsson, and J. A. Hertz, 1990, in *Advances in Neural Information Processing Systems II,* edited by D. S. Touretzky (Morgan Kaufmann, San Mateo), p. 733.
Kühn, R., S. Bös, and J. L. van Hemmen, 1991, Phys. Rev. A **43**, 2084.
Kurchan, J., and E. Domany, 1990, J. Phys. A **23**, L847.
Langevin, P., 1908, Compt. Rend. **43**, 530.
Le Cun, Y., 1986, in *Disordered Systems and Biological Organization,* edited by E. Bienenstock, F. Fogelman-Soulié, and G. Weisbuch (Springer, Berlin), p. 234.
Le Cun, Y., J. S. Denker, and S. A. Solla, 1990, in *Advances in Neural Information Processing Systems II,* edited by D. S. Touretzky (Morgan Kaufmann, San Mateo), p. 598.
Le Cun, Y., I. Kanter, and S. A.Solla, 1991, Phys. Rev. Lett. **66**, 2396.
Levin, E., N. Tishby, and S. A. Solla, 1989, in *Proceedings of the 2nd Workshop on Computational Learning Theory* (Morgan Kaufmann, San Mateo).
Levin, E., Y. Le Cun, and V. Vapnik, 1992, "Measuring the capacity of a learning machine (II)," AT&T Bell Laboratories, preprint.
Lippman, R. P., 1989, Neural Comput. **1**, 1.
MacKay, D. J. C., 1992, Neural Comput. **4**, 448.
Makram-Ebeid, S., J.-A. Sirat, and J.-R. Viala, 1989, in *International Joint Conference on Neural Networks* (IEEE, New York), Vol. II, p. 373.
Marchand, M., M. Golea, and P. Ruján, 1990, Europhys. Lett. **11**, 487.
Martinez, D., and D. Esteve, 1992, Europhys. Lett. **18**, 95.
Mato, G., and N. Parga, 1992, J. Phys. A **25**, 5067.
McCulloch, W. S., and W. Pitts, 1943, Bull. Math. Biophys. **5**, 115.
Meir, R., and J. F. Fontanari, 1992, Phys. Rev. A **45**, 8874.
Mézard, M., and J.-P. Nadal, 1989, J. Phys. A **22**, 2191.
Mézard, M., J.-P. Nadal, and G. Toulouse, 1986, J. Phys. (France) **47**, 1457.
Mézard, M., G. Parisi, and M. A. Virasoro, 1987, *Spin Glass Theory and Beyond* (World Scientific, Singapore).
Mézard, M., and S. Patarnello, 1989, "On the capacity of feedforward layered networks," Ecole Normale Supérieure Preprint No. LPTENS 89/24.
Minsky, M. L., and S. A. Papert, 1969, *Perceptrons* (MIT, Cambridge, MA).
Mitchell, T. M., 1982, Artif. Intell. **18**, 203.
Mitchison, G. J., and R. M. Durbin, 1989, Biol. Cybern. **60**, 345.
Moody, J., 1992, "Generalization, weight decay and architecture selection for nonlinear learning schemes," Yale University preprint.
Müller, B., and J. Reinhardt, 1991, *Neural Networks—An Introduction* (Springer, Berlin).
Nabutovsky, D., and E. Domany, 1991, Neural Comput. **3**, 604.
Nabutovsky, D., T. Grossman, and E. Domany, 1990, Complex Syst. **4**, 519.
Nadal, J.-P. 1989, Int. J. Neural Syst. **1**, 55.
Nadal, J.-P., and A. Rau, 1991, J. Phys. I France **1**, 1109.
Onsager, L., 1944, Phys. Rev. **65**, 117.
Opper, M., 1989, Europhys. Lett. **8**, 389.
Opper, M., S. Diederich, and J. K. Anlauf, 1988, in *Neural Networks from Models to Applications,* edited by L. Personnaz and G. Dreyfus (I.D.S.E.T., Paris).
Opper, M., and D. Haussler, 1991a, Phys. Rev. Lett. **66**, 2677.
Opper, M., and D. Haussler, 1991b, in *Proceedings of the 4th Annual Workshop on Computational Learning Theory,* edited by M. K. Warmuth and L. G. Vallian (Morgan Kaufmann, San Mateo).
Opper, M., W. Kinzel, J. Kleinz, and R. Nehl, 1990, J. Phys. A **23**, L581.
Parisi, G., 1988, *Statistical Field Theory* (Addison-Wesley, Redwood City, CA).
Parisi, G., and F. Slanina, 1992, Europhys. Lett. **17**, 497.
Parrando, J. H. R., and C. Van den Broeck, 1992, "Vapnik Chervonenkis bounds for generalization," University of California, San Diego, preprint.
Patel, H. K., 1992, "Computational complexity, learning, rules and storage capacities: A Monte Carlo study for the binary perceptron," Universität Heidelberg preprint.
Patarnello, S., and P. Carnevali, 1987, Europhys. Lett. **4**, 503.
Penney, R. W., and A. Wendenmuth, 1992, private communication.
Peretto, P., 1984, Biol. Cybern. **50**, 51.
Peretto, P., 1991, *The Modeling of Neural Networks* (Cambridge University, Cambridge, England).
Personnaz, L., I. Guyon, and G. Dreyfus, 1986, Phys. Rev. A **34**, 4217.
Plaut, D., S. Nowlan, and G. Hinton, 1986, "Experiments on learning by back propagation," Technical Report No. CMU-CS-86-126 (Department of Computer Science, Carnegie-Mellon University, Pittsburgh, PA).
Ritter, H., K. Schulten, and T. Martinetz, 1990, *Neuronale Netze* (Addison-Wesley, Bonn).
Rohwer, R., 1990, in *Advances in Neural Information Processing Systems II,* edited by D. S. Touretzky (Morgan Kaufmann, San Mateo), p. 558.
Rosenblatt, F., 1962, *Principles of Neurodynamics* (Spartan, New York).
Ruján, P., and M. Merchand, 1989, Complex Syst. **3**, 229.
Rumelhart, D. E., and J. L. McClelland, 1986, Eds., *Parallel Distributed Processing* (MIT, Cambridge, MA).
Saad, D., and E. Marom, 1990, Complex Syst. **4**, 107.
Schmitz, H. J., G. Pöppel, F. Wünsch, and U. Krey, 1990, J. Phys. (France) **51**, 167.
Schwartz, D. B., V. K. Samalam, S. A. Solla, and J. S. Denker, 1990, Neural Comput. **2**, 371.
Schwarze, H., 1991, Diplomarbeit (Universität Giessen).
Schwarze, H. and J. Hertz, 1992a, Europhys. Lett. **20**, 375.
Schwarze, H., and J. Hertz, 1992b, "Generalization in a fully-connected committee machine," Nordita Preprint No. 92/61 S.
Schwarze, H., M. Opper, and W. Kinzel, 1992, Phys. Rev. A **46**, R6185.
Sejnowski, T. J., and C. R. Rosenberg, 1987, Complex Syst. **1**, 145.
Seung, H. S., 1991, private communication.
Seung, H. S., M. Opper, and H. Sompolinsky, 1992, "Query by committee," *Proceedings of the 5th Annual ACM Workshop on Computational Learning Theory* (ACM, NY).
Seung, H. S., H. Sompolinsky, and N. Tishby, 1992, Phys. Rev. A **45**, 6056.
Sherrington, D., 1986, Prog. Theor. Phys. **87**, 180.
Sherrington, D., and S. Kirkpatrick, 1975, Phys. Rev. Lett. **35**, 1792.
Sietsma, J., and R. J. F. Dow, 1988, in *IEEE International Joint Conference on Neural Networks,* Vol. I (IEEE, New York), p. 325.
Sirat, J.-A., and J.-P. Nadal, 1990, Network **1**, 423.
Sjöberg, L. Ljung, 1992, "Overtraining, regularization and searching for minimum in neural networks," Linköping Uni-

versity, Sweden, preprint.
Solla, S. A., 1992, private communication.
Solla, S. A., E. Levin, and M. Fleischer, 1988, Complex Syst. **2**, 625.
Sompolinsky, H., and N. Tishby, 1990, Europhys. Lett. **13**, 567.
Sompolinsky, H., N. Tishby, and H. S. Seung, 1990, Phys. Rev. Lett. **65**, 1683.
Sourlas, N., 1989, Nature **339**, 693.
Stanley, H. E., 1971, *Introduction to Phase Transitions and Critical Phenomena* (Oxford University Press, Oxford).
Tishby, N., E. Levin, and S. Solla, 1989, in *IEEE International Joint Conference on Neural Networks,* Vol. II (IEEE, New York), p. 403.
Utans, J., and J. Moody, 1991, in *Proceedings of the First International Conference on Artificial Intelligence Applications on Wall Street* (IEEE Computer Society, Los Alamos, CA).
Vallet, F., 1989, Europhys. Lett. **8**, 747.
Vallet, F., J. Cailton, and P. Refregier, 1989, Europhys. Lett. **9**, 315.
Van den Broeck, C., and R. Kawai, 1990, Phys. Rev. A **42**, 6210.
van Hemmen, J. L., D. Grensing, A. Huber, and R. Kühn, 1988, J. Stat. Phys. **50**, 231.
van Hemmen, J. L., G. Keller, and R. Kühn, 1988, Europhys. Lett. **5**, 663.
Vapnik, V. N., 1982, *Estimation of Dependences Based on Empirical Data* (Springer, Berlin).
Vapnik, V. N., 1992, "Measuring the capacity of a learning machine (I)," AT&T Bell Laboratories preprint.
Vapnik, V. N., and A. Y. Chervonenkis, 1971, Theory Probab. Its Appl. **16**, 264.
Venkatesh, S., 1986, in *Neural Networks for Computing,* AIP Conference Proceedings No. 151, edited by J. S. Denker (AIP, New York), p. 440.
Vogl, T. P., W. T. Zink, J. K. Mangis, D. L. Alkon, and A. K. Rigler, 1988, Biol. Cybern. **59**, 257.
von Lehmann, A., E. G. Paek, P. F. Iao, A. Marrakchi, and J. S. Patel, 1988, in IEEE *International Conference on Neural Networks,* Vol. I (IEEE, New York), p. 335.
Waibel, A., T. Hanazawa, G. Hinton, K. Shikano, and K. J. Lang, 1989, IEEE Trans. Acoust. Speech Signal Process. **37**, 328.
Watkin, T. L. H., 1991, unpublished.
Watkin, T. L. H., 1993, "Optimal learning with a neural network," Europhys. Lett. **21**, 871.
Watkin, T. L. H., and A. Rau, 1992a, J. Phys. A **25**, 113.
Watkin, T. L. H., and A. Rau, 1992b, Phys. Rev. A **45**, 4102.
Watkin, T. L. H., A. Rau, D. Bollé, and J. van Mourik, 1992, J. Phys. I France **2**, 167.
Watkin, T. L. H., and D. Sherrington, 1992, "An exactly solved neural network which is susceptible to simulation," Oxford University preprint.
Watrous, R. L., 1987, in *IEEE 1st International Conference on Neural Networks,* edited by M. Caudill and C. Butler (IEEE, New York), Vol. II, p. 619.
Weisbuch, G., 1991, *Complex Systems Dynamics* (Addison-Wesley, Redwood City, CA).
Wendemuth, A., 1991, Diplomarbeit (Universität Giessen).
Widrow, B., and M. E. Hoff, 1960, in *1960 IRE WESCON Conv. Rep. IV* (Institute of Radio Engineers, New York), p. 96.
Winder, R. O., 1963, IEEE Trans. Electron. Comput. **EC-12**, 561.
Wittner, B. S., and J. S. Denker, 1988, in *Neural Information Processing Systems,* edited by D. Z. Anderson (AIP, New York), p. 850.
Wong, K. Y. M., and D. Sherrington, 1990, J. Phys. A **23**, 4659.
Wu, F. Y., 1982, Rev. Mod. Phys. **54**, 235.
Zollner, R. H. J. Schmitz, F. Wünsch, and U. Krey, 1992, Neural Networks **5**, 771.